INAUGURAL - DISSERTATION

zur

Erlangung der Doktorwürde

der

Naturwissenschaftlich-Mathematischen

Gesamtfakultät

der

Ruprecht-Karls-Universität

Heidelberg

Vorgelegt von

Dipl.-Geogr. Nicolai Freiwald

aus

Heidelberg

Tag der mündlichen Prüfung: 06.07.2007

Interaktives, webbasiertes 3D-Informationssystem für den Heidelberger Universitätscampus

Gutachter:

Prof. Dr. Peter Meusburger

Prof. Dr. Rüdiger Glaser

Zusammenfassung:

Die Arbeit thematisiert das bauliche Abbild und die Entwicklung des Heidelberger Universitätscampus. Im Mittelpunkt stehen zum einen die Schaffung eines virtuellen, dreidimensionalen Computermodells des Untersuchungsgebiets und zum anderen die Konzeption und Implementierung eines raum-zeitlichen Informationssystems, das den Universitätscampus zum Inhalt hat. Neben den drei Dimensionen des Raums wird als vierte Dimension die Zeit integriert. Dies erlaubt die Abbildung zukünftiger und historischer Planungen. Das 3D-Modell basiert auf aktuellen digitalen und historischen analogen Quellen sowie auf umfangreichen eigenen Erhebungen. Die digitale Prozesskette zur Erfassung, Verarbeitung und Verwaltung der Daten ist so konzipiert, dass eine möglichst breite Verwendung der Inhalte gewährleistet ist. Für die Schaffung des 3D-Modells wird ein Konzept verfolgt, welches dessen Einsatz für unterschiedlichste Anwendungszwecke gestattet. Das raum-zeitliche Informationssystem ist so gestaltet, dass dem Nutzer ein ubiquitärer, interaktiver Zugriff auf das 3D-Modell und den damit verknüpften thematischen Daten ermöglicht wird. Das Informationssystem wird über das Internet bereitgestellt und besitzt den immersiven Charakter einer Virtual Reality Umgebung. Der funktionale und inhaltliche Umfang des Systems bezieht formell in den städtebaulichen Planungsprozess involvierte Akteure wie Stadtplaner und Architekten ebenso ein wie die an Planungspartizipation und allgemeinen, raumbezogenen Informationen interessierte breite Öffentlichkeit. Die Implementierung des interaktiven, webbasierten 3D-Informationssystems basiert auf international standardisierten, nicht-proprietären Softwaretechnologien.

Abstract:

The subject of this thesis is the constructional image and development of the Heidelberg University campus. Its focus is the creation of a virtual three dimensional computer model of the project area on the one hand and the conceptual design as well as the implementation of a spatiotemporal information system centering on the university campus on the other hand. Apart from the three dimensions of space the fourth dimension of time is integrated, allowing the portrayal of future as well as historical planning projects. The 3D model is based on current digital and historical analogue sources as well as own comprehensive studies. The digital workflow for the compilation, processing and administration of the data is designed so as to ensure a widespread application of its contents. The conceptual design of the 3D model makes it apt for most diversified application purposes. The spatiotemporal information system allows the user ubiquitous, interactive access to the 3D model and the thematic data associated therewith. The information system is made available through the internet and has the immersive character of a virtual reality environment. The functional and conceptual scope of the system incorporates specialized urban planning process personnel such as urban developers and architects as well as the general public interested in planning participation and general spatial information. The implementation of the interactive, web-based 3D information system is based on international standardized, non-proprietary software technology.

Danksagung

Danken möchte ich Herrn Prof. Dr. Peter Meusburger für die Betreuung dieser Dissertation und für die Gewährung des notwendigen Freiraums, das Projekt nach meinen Vorstellungen umsetzen zu können.

Dem Universitätsbauamt Heidelberg danke ich für die freundliche und unbürokratische Bereitstellung von Daten.

Ein besonderer Dank gilt meiner Familie für Geduld und Unterstützung in jeder Hinsicht.

Gefördert mit Mitteln der Stadt-Heidelberg-Stiftung

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1 Einleitung

„Ein Bild sagt mehr als tausend Worte.“ Obwohl es sich bei diesem Sprichwort nicht, wie häufig angenommen, um eine Jahrtausende alte chinesische Spruchweisheit handelt, sondern eher profan um den Werbeslogan eines nordamerikanischen Werbefachmanns der 1920er Jahre (MIEDER 1989, 25 ff.), büßt die Aussage selbst nicht an Wahrheit ein.

Die mediale Präsenz von Bildern war selten so ausgeprägt wie heute. Synergieeffekte von Fotografie, Massenmedien und Computertechnik haben den Einfluss und die Verwendung bildhafter Darstellungen auf bzw. in Gesellschaft, Kultur und Wissenschaft gesteigert. So aussagekräftig und informationsträchtig Bilder auch sein können, ein erheblicher Nachteil bleibt für gewisse Anwendungszwecke ihr statischer und auf zwei Dimensionen beschränkter Charakter.
Zwar können bewegte Bilder in Form von Filmen oder Animationen eine weitere Raumdimension sowie die Dimension Zeit vermitteln, jedoch transportieren sie genauso wie statische Bilder nur unveränderliche und in ihren Eigenschaften bzw. Inhalten vom Betrachter nicht beeinflussbare Informationen. Bilder, Filme und Animationen sind als Visualisierung somit grundsätzlich nicht geeignet, auf eine Reaktion des Betrachters einzugehen. Um die Aussagekraft von Visualisierungen zu steigern, benötigt der Betrachter eine Möglichkeit, mit ihnen zu interagieren und direkt auf die zugrunde liegenden Inhalte zuzugreifen. Dadurch wird aus dem Betrachter ein Benutzer, der Bildinhalte entsprechend seinen Vorstellungen oder Ansprüchen verändern und anpassen sowie weitere Inhalte hinzufügen kann. So entsteht ein Dialog, der den ursprünglich einseitig rezipierenden Charakter einer Bild-Betrachter-Konstellation in Bezug auf die Informationsvermittlung qualitativ und quantitativ weit übertreffen kann.

Interaktive 3D-Informationssysteme können die Basis für einen so gearteten Dialog bieten. Sie stellen dem Benutzer Inhalte, zum Beispiel mehrdimensionale Geodaten

sowie Funktionalitäten zu deren Erschließung zur Verfügung. Um den Bezug zu eingangs aufgeführtem Sprichwort herzustellen, versetzen solche Informationssysteme den Nutzer in die Lage, sich selbst ein Bild von etwas zu machen. Diese Art von Informationssystemen scheint prädestiniert zur Anwendung in denjenigen Wissenschaftsdisziplinen, die den Raum und dessen immanente Prozesse thematisieren; den Geowissenschaften. Sie können dort der internen Wissensgenerierung und vor allem der externen Informationsvermittlung dienen.

Mit der Einführung des virtuellen Online-Globus Google Earth steht seit einigen Monaten eine Plattform im Internet bereit, welche die Informationstiefe dreidimensionaler globaler Geodaten erstmals für die breite Öffentlichkeit erfahrbar macht. Auch wenn die gewaltige Medienpräsenz von Google Earth und die Nonchalance, mit der die Online-Welt diese Plattform inzwischen für die verschiedensten Zwecke nutzt, anderes vermuten lässt, so besteht in den angesprochenen Wissenschaftsdisziplinen erheblicher Forschungsbedarf bezüglich der Erhebung, der Verwaltung, der Analyse und der Präsentation von dreidimensionalen Geodaten. Vorliegende Arbeit leistet einen Beitrag, um diesbezüglich bestehende Forschungslücken zu schließen. Bezogen auf die digitale raum-zeitliche Abbildung der Stadt Heidelberg wird ebenfalls eine Lücke geschlossen.

Die Arbeit ist in sieben Kapitel gegliedert. Die ersten drei Kapitel dienen der Definition und Abgrenzung der Forschungsziele, einer Einführung in die übergeordnete Thematik sowie einem Abriss der Entwicklung des Untersuchungsgebiets. Die Kapitel vier, fünf und sechs widmen sich der Bearbeitung der Forschungsfragen und -ziele sowie einer Präsentation der Ergebnisse. Das letzte Kapitel schließt die Arbeit mit einer zusammenfassenden Betrachtung ab.

Der Arbeit liegt im Anhang eine CD bei, die die wichtigsten Forschungsergebnisse anhand zweier Filme präsentiert.

2 Forschungsziele und Einordnung der Dissertation

Die Dissertationsschrift thematisiert das bauliche Abbild und die Entwicklung des Heidelberger Universitätscampus. Im Mittelpunkt der Arbeit stehen Konzeption und Implementierung eines raum-zeitlichen Informationssystems, das den Universitätscampus zum Inhalt hat.

2.1 Forschungsziele

Vorliegende Dissertationsschrift ist in der Geographie angesiedelt. Es werden Forschungsfragen aus unterschiedlichen Teilbereichen dieser Disziplin sowie benachbarter Wissenschaften behandelt. Die Geographie spannt dabei einen Rahmen zwischen Raumplanung, historischer Stadtentwicklung, Architektur und Informatik auf.

Die Arbeit verfolgt vier Forschungsziele:

1. Digitale Abbildung von planerischen Prozessen und Inhalten
2. Schaffung eines digitalen mehrdimensionalen Abbilds des Universitätscampus
3. Konzeptioneller Entwurf des Informationssystems
4. Implementierung des Informationssystems

Nachfolgend werden die einzelnen Forschungsziele erläutert und abgegrenzt.

2.1.1 Digitale Abbildung von planerischen Prozessen und Inhalten

Das digitale Informationssystem soll unter anderem über Aspekte der zukünftigen baulichen Entwicklung des Universitätscampus informieren sowie Funktionen anbieten, die den Planungsprozess aktiv unterstützen können. Hierzu werden Instrumente und Inhalte der Planung auf ihre digitale Abbildbarkeit hin geprüft werden.

Ziel der Arbeit ist es nicht, sämtliche mit dem Heidelberger Universitätscampus im Zusammenhang stehenden baulichen Planungsprozesse zu analysieren. Da es sich um Hochschulbauten handelt, überschneiden sich kommunale, staatliche und Kompetenzen des Bundes. Die Vernetzung von Hochschulplanung und Stadtplanung sowie der Kompetenzbereich der verschiedenen Akteure wurden bereits von FREIWALD (2003) beschrieben. Die Vielzahl der involvierten Akteure und Planungsebenen sowie die Vielfalt wirkender Planungsprozesse machen eine Abgrenzung notwendig.

Die Arbeit beschränkt sich darauf, bestimmte Teilprozesse der Bauleit- und Objektplanung aufzugreifen. Für diese werden Funktionalitäten bzw. Methoden entworfen, die den jeweiligen Teilprozess informell unterstützen können.
Zum einen soll auf diese Weise das formelle Planungsinstrument der Öffentlichkeitsbeteiligung digital abgebildet werden. Im Fokus steht dabei eine für den Bürger nachvollziehbare Vermittlung komplexer Planungsinhalte mithilfe dreidimensionaler Visualisierung.
Zum anderen soll für einzelne Werkzeuge auf der Ebene bestimmter Planungsakteure eine Möglichkeit zur digitalen Repräsentation gefunden werden. Hierbei soll der Schwerpunkt auf informellen Teilen des Planungsprozesses liegen. Teile des Konzeptes analoger Architekturmodelle sollen digital zur Verfügung gestellt werden.

Als bauliche Inhalte aktueller und zukünftiger Planungen sollen Objekte aufgegriffen werden, die dazu beitragen werden, das derzeitige Bild des Universitätscampus nachhaltig zu verändern.

2.1.2 Schaffung eines digitalen mehrdimensionalen Abbilds des Universitätscampus

Den Kern des digitalen Informationssystems soll ein dreidimensionales, digitales Modell des Heidelberger Universitätscampus bilden.
Dieses Modell soll nicht nur den gegenwärtigen Baubestand abbilden, sondern auch im Rahmen aktueller Planungen projektierte Bauten erfassen. Zukünftige Planungen bzw. Planungsalternativen und deren Entwurfsvarianten sollen ebenfalls in das dreidimensionale Modell Eingang finden. Neben gegenwärtigem und zukünftigem Baubestand soll untersucht werden, inwieweit nicht realisierte historische Planungen wesentlichen Einfluss auf das heutige Erscheinungsbild gehabt haben könnten. Diese historischen Ent-

würfe sollen auch als virtuelles 3D-Modell umgesetzt werden. Die Abbildung historischer und zukünftiger Planungen bzw. möglicher Varianten integriert die Zeit als vierte Dimension in das Informationssystem.
Neben den baulichen Ressourcen sollen auch die verbindende Infrastruktur sowie prägnante Geländemerkmale des Untersuchungsgebiets digital erfasst und als eigenes 3D-Modell zur Verfügung gestellt werden.
Das 3D-Modell sowie die Prozesskette von Erfassung und Weiterverarbeitung der dem 3D-Modell zu Grunde liegenden Daten sollen so gestaltet werden, dass eine spezifische Anpassung an eine bestimmte Nutzungsart erst zu einem möglichst späten Punkt in der Prozesskette notwendig wird. Damit soll eine vielseitige Verwendungsmöglichkeit des 3D-Modells sowie der zu Grunde liegenden Daten sicher gestellt werden.
Die virtuelle dreidimensionale Modellierung des aktuellen Baubestandes soll lückenlos und vollständig realisiert werden. Für die dreidimensionale Abbildung zukünftiger und historischer Planungen wird eine Auswahl getroffen.

Mit diesem Forschungsziel soll eine weitere Lücke geschlossen werden, die sich auf die Stadt Heidelberg bezieht. Wissenschaftliche Arbeiten mit dem Fokus auf der digitalen Darstellung und Erschließung Heidelbergs sind bislang kaum vorhanden. So existiert derzeit weder für Heidelberg noch für einen seiner Stadtteile, als solcher wird der Universitätscampus angesehen, ein flächendeckendes detailliertes 3D-Computermodell, das auch historische oder zukünftige Bauten enthält und ohne Einschränkungen für eine interaktive webbasierte Darstellung geeignet ist.
Die Arbeiten von JÖST (2000) und ZIPF (2000) beschäftigen sich mit der Konzipierung und prototypischen Umsetzung eines räumlichen Touristeninformationssystems für Heidelberg. Die digitale Darstellung der Stadt beschränkt sich in diesen Arbeiten überwiegend auf zweidimensionale Karten im Rahmen eines webbasierten Geographischen Informationssystems. SCHILLING (2002) nutzt dreidimensionale Darstellungen von Heidelberg zur Visualisierung von Routen. Da das Ziel der Arbeit auf der dynamischen Integration von 2D- und 3D-Daten in einem Geographischen Informationssystem liegt, bleibt die dreidimensionale Darstellung auf einem geringen Detaillierungsniveau; die Zeit als vierte Dimension spielt zudem keine Rolle. Für die Arbeit von JANY (2005) wurden 3D-Modelle angefertigt, die punktuell für bestimmte Teile der Altstadt deren historisches Erscheinungsbild zeigen. Eine flächenhafte und zusammenhängende Darstellung findet jedoch nicht statt. Die Modelle sind zudem für eine interaktive Online-Darstellung nur wenig geeignet.

2.1.3 Konzeptioneller Entwurf des Informationssystems

Die Informationsplattform soll es dem Benutzer erlauben, sich über Baubestand, historische Pläne und zukünftige Vorhaben im Untersuchungsgebiet zu informieren. Kern des Informationssystems soll das mehrdimensionale Abbild des Universitätscampus sein.

Inhaltlich soll die Plattform im Wesentlichen zwei Ansprüchen genügen. Zum einen soll sie als allgemeines 3D-Informationssystem dienen, welches über Lage, Funktion und Erreichbarkeit der universitären Gebäude Auskunft gibt. Neben solchen unmittelbar raumbezogenen Inhalten soll auch der Zugriff auf thematische Information möglich sein. Hierzu soll die dreidimensionale Geometrie mit entsprechenden Sachdaten verknüpft werden. Zum anderen soll die Plattform dazu dienen, bestimmte, im Forschungsziel 1 definierte Aspekte der baulichen Planung zu unterstützen. Um dies zu erreichen, wird das 3D-Informationsystem um die Dimension Zeit erweitert, damit Veränderungen des Baubestandes visualisiert werden können. Auch der planungstechnische Teil der Plattform soll eine thematische Ebene mit Sachdaten zu baulichen Vorhaben integrieren. Darüber hinaus sollen bestimmte Funktionalitäten verfügbar sein, die formelle Akteure bei ihrer Arbeit unterstützen können. Die Zielgruppe der Informationsplattform umfasst somit neben der breiten Öffentlichkeit, Bürger, die am Planungsprozess partizipieren möchten, sowie in formelle Planungsprozesse involvierte Akteure wie Stadtplaner oder Architekten.
Technisch soll die Plattform so konzipiert werden, dass ein nahezu standortunabhängiger Zugriff ermöglicht wird. Dies soll durch die Wahl des Mediums Internet gewährleistet werden. Die Plattform soll so beschaffen sein, dass dem Benutzer die Informationen interaktiv, dynamisch und dreidimensional zur Verfügung stehen. Dies soll durch die Einbettung des 3D-Computermodells in eine diese Voraussetzung erfüllende technische Umgebung erfolgen. Eine Herausforderung besteht hierbei unter anderem in der Integrierung der dreidimensionalen Daten in einer Form, die Interaktionen in Echtzeit, d.h. ohne merkliche Verzögerung, erlaubt. Die Konzeption soll so weit wie möglich auf international anerkannten Standardtechnologien basieren. Durch die offen zugängliche Spezifikation solcher Standards soll das System für jedermann funktional erweiterbar sein. Standards beugen darüber hinaus einer Abhängigkeit von bestimmten Herstellern vor und ermöglichen einen Einsatz frei von Lizenzgebühren. Die Architektur des Systems soll so gestaltet sein, dass der Nutzer es um eigene Inhalte erweitern kann.

2.1.4 Implementierung des Informationssystems

Auf Basis der in der Konzeption postulierten Anforderungen an ein interaktives, webbasiertes Informationssystem sollen entsprechende Softwarekomponenten entwickelt bzw. programmiert werden.
Ziel ist die Realisierung einer browserbasierten Plattform, die Kernfunktionalitäten wie die Navigation im 3D-Modell, die Interaktion mit dem Modell sowie die Verknüpfung des Modells mit Sachdaten bereitstellt. Die Funktionsfähigkeit und Interneteignung des Informationssystems sollen online demonstriert werden können.

Das Erreichen dieses Forschungsziels soll dazu beitragen, bestehende Forschungslücken zumindest zum Teil zu schließen. So existiert derzeit kein 3D-Informationssystem, welches in gleicher Weise auf internationalen Standards aufbaut, Instrumente der Planung abbildet und die interaktive, browserbasierte Online-Visualisierung großflächiger, detaillierter Baubestände in Kombination mit zukünftiger und historischer Bausubstanz erlaubt.
Frei verfügbare 3D-Informationssysteme wie Google Earth oder World Wind, die ebenfalls einen webbasierten Zugriff auf 3D-Geodaten erlauben, basieren nicht auf Standards und sind in erster Linie für kleinmaßstäbliche Geodaten konzipiert; zudem ist eine direkte Verknüpfung der 3D-Inhalte mit thematischen Daten nicht gegeben. Beide Systeme setzen überdies die manuelle Installation einer umfangreichen Software voraus und bieten nur eingeschränkte Navigations- und Interaktionsmöglichkeiten (WILK 2007, 69). Kommerzielle Systeme wie beispielsweise TerrainView (STEIDLER & BECK 2005, 221 ff.), die dediziert für die Darstellung von 3D-Stadtmodellen ausgelegt sind, erlauben zwar die Integration standardisierter Datenformate, beruhen aber selbst nicht auf solchen Standards. Diese Systeme können zwar zur Darstellung Daten aus dem Internet laden, eine vollständige Integration in das Hyperreferenz-Konzept des Internets fehlt jedoch. Die manuelle Installation eines umfangreichen Softwarepakets ist auch hier notwendig. Wissenschaftliche Projekte, in denen die Visualisierung von 3D-Geodaten thematisiert wird, verfolgten bislang andere Ziele als die vorliegende Arbeit. Das Projekt GDI3D (Geodateninfrastruktur für 3D-Geodaten) der Fachhochschule Mainz beispielsweise hat im Kern den Aufbau einer Infrastruktur für dreidimensionale Geodaten zum Ziel (ZIPF 2007). Teil des Vorhabens ist aber auch die Entwicklung einer Visualisierungssoftware für 3D-Stadtmodelle. Aufgrund der Funktionsanforderungen basiert diese Software jedoch bewusst auf nicht standardisierten

Technologien wie Java bzw. Java3D. Für die Datenhaltung wird jedoch ein Modell evaluiert, welches zukünftig als Standard spezifiziert werden könnte.

2.2 Untersuchungsgebiet und Untersuchungszeitraum

Die Einrichtungen der Universität Heidelberg konzentrieren sich nicht auf einen einzigen Standort innerhalb des Stadtgebiets. Geistes- und Sozialwissenschaften liegen südlich des Neckars verteilt in der Heidelberger Altstadt. Natur- und Sportwissenschaften sowie die größten Teile der Medizin befinden sich im Neuenheimer Feld, welches nördlich des Neckars im Stadtteil Neuenheim liegt.
Während die Universitätseinrichtungen im Bereich der Altstadt durch ihre Streulage gekennzeichnet sind, sind die universitären Bauten im Neuenheimer Feld in einer Art und Weise räumlich konzentriert, die es erlauben, von einem Universitätscampus zu sprechen.
Der Teil Neuenheims, welcher sich zwischen Berliner Straße im Osten und Neckar im Süden bzw. Westen befindet, trägt den Namen „Neuenheimer Feld“. Der Bereich, in dem sich die universitären Einrichtungen befinden wird als „Im Neuenheimer Feld“ (INF) bezeichnet. Dieses Areal repräsentiert das Untersuchungsgebiet.
Gegenstand der Untersuchung sind neben den Universitätsgebäuden auch die baulichen Ressourcen mit der Universität assoziierter Einrichtungen. Dies sind beispielsweise die Max-Planck-Institute, das Deutsche Krebsforschungszentrum oder der Technologiepark. Neben Grünflächen, Gehwegen und Parkflächen wird auch die den Universitätscampus erschließende Verkehrsinfrastruktur thematisiert.

Die digitale Abbildung des Untersuchungsgebiets wird sich nicht auf eine reine Darstellung des Bestandes beschränken, sondern soll die zeitliche Entwicklung dieses Bestandes visualisieren können. Neben historischen, niemals ausgeführten Planungen werden zukünftige kurz- bis mittelfristige Vorhaben sowie langfristige Planungsideen aufgegriffen und als digitales dreidimensionales Modell abgebildet. Die Zeit als vierte Dimension spielt somit eine wesentliche Rolle in dieser Arbeit.
Der Untersuchungszeitraum erstreckt sich über einen Bereich von mehr als 100 Jahren. Die erste universitäre Einrichtung im Neuenheimer Feld war der 1913 dorthin verlegte Botanische Garten (LURZ & VOGT 1990, 90 ff.). Aktuelle Vorhaben wie das des so genannten Heidelberger Klinikringes oder seit langem diskutierte Ideen wie eine mögliche fünfte Neckarquerung werden erst mittel- bis langfristig realisiert werden bzw. zu

konkreten Planungen führen (UNIVERSITÄTSKLINIKUM HEIDELBERG PLANUNGSGRUPPE MEDIZIN 2005; STADT HEIDELBERG 2002; UNIVERSITÄTSBAUAMT HEIDELBERG 2002).

2.3 Einordnung und theoretischer Kontext

Dieser Untersuchung liegt eine interdisziplinäre Herangehensweise zu Grunde. Einsichten, Erkenntnisse und methodische Ansätze aus Geographie, Planungswissenschaft, Architektur, Geschichte, Informatik und Computergrafik werden unter dem vereinenden Dach einer angewandten geographischen Betrachtungsweise bearbeitet.
Im Folgenden werden diejenigen Konzepte aus den genannten Disziplinen dargestellt, welche für die vorliegende Dissertationsschrift relevant sind. Um eine Einordnung der Arbeit zu erleichtern, wird punktuell der Bezug zu den einzelnen Forschungszielen hergestellt. Hiermit soll ein grobes Netz aufgespannt werden, an dessen Knotenpunkten vorliegende Arbeit konzeptuell anknüpft.

2.3.1 Geographische Informationssysteme

Geographische Informationssysteme, abkürzend auch als GIS bezeichnet, haben sich während der vergangenen zwei Jahrzehnte zu einem wichtigen Arbeitswerkzeug derjenigen Wissenschaftsdisziplinen entwickelt, die den Raum als solchen und die ihm innewohnenden Prozesse erforschen. Eine allgemeingültige Definition für ein GIS, welche Aufgaben es hat oder aus welchen Komponenten es besteht, existiert bislang nicht (CHRISTIANSEN & ERB 2002).
Nach BILL (1999, 5 ff.) besteht ein GIS aus den Elementen Anwender, Hardware, Software und Daten. Wesentliche Werkzeuge eines Geographischen Informationssystems sind dabei Instrumente zur Datenmodellierung und Analyse der Daten, deren Ergebnisse der Entscheidungsfindung dienen sollen. Insgesamt bietet ein GIS vier funktionale Komponenten. Dies sind die Module Erfassung, Verwaltung, Analyse und Präsentation.

Die Daten, zu deren Modellierung und Verwaltung ein GIS dient, werden zusammenfassend als Geodaten bezeichnet. Zudem findet eine weitere Gliederung in Geometrie- und Sachdaten statt. Während Geometriedaten Informationen über Lage und Form eines Objektes auf der Erdoberfläche enthalten, können mittels Sachdaten Eigenschaften eines Objektes beschrieben werden (LIEBIG 1999, 15 ff.).

Ein Mehrwert von Geographischen Informationssystemen gegenüber reinen CAD- oder Zeichenprogrammen bzw. gegenüber Datenbanken besteht in der explizit raumbezogenen Abbildung und Verknüpfung von Geometrie- und Sachdaten. GIS als Konzept bietet insgesamt vielfältige Vorteile, da unterschiedliche Informationsebenen und Inhalte aus verschiedenen Quellen miteinander verknüpft werden können und so eine integrierte Informationsverwaltung, -analyse und -vermittlung ermöglicht wird (BARTELME 2005, 11 ff.).
Die Möglichkeit zur Bearbeitung zweidimensionaler Geodaten gehört bei den aktuellen GIS-Applikationen standardmäßig zum Funktionsumfang. Allerdings erfordern bestimmte Aufgabenbereiche, wie sie zum Beispiel in der Stadtplanung vorkommen, die Möglichkeit zur Bearbeitung dreidimensionaler Geometriedaten (LIEBSCHER 2002, 344 ff.). Einfache Instrumente zur Analyse dreidimensionaler geometrischer Daten sind bereits in den GIS-Paketen der großen Hersteller enthalten. Damit sind beispielsweise Sichtfeld- und Expositionsanalysen dreidimensionaler Geländemodelle möglich (FREIWALD ET AL. 2006, 37 ff.). Komplexere Aufgabenbereiche wie Schallausbreitung zur Erstellung von Lärmschutzkatastern oder die Ausbreitung von elektromagnetischer Strahlung zur Planung von Funknetzen bereiten geographischen Informationssystemen jedoch noch Probleme bzw. sind nur mit Zusatzprogrammen zu bewältigen (MÜLLER 2005, 392 ff.; PFÄFFLIN ET AL. 2004, 518 ff.). Bei der Präsentation von Geodaten, der vierten GIS-Komponente, sind Geographische Informationssysteme insbesondere bei 3D-Geodaten speziellen Programmen für 3D-Modellierung und Animation weit unterlegen (GÖBEL 2006). Eine umfassende Abhandlung des Themas 3D-GIS bieten COORS & ZIPF (2005).

Geographische Informationssysteme sind Softwareapplikationen, die eine Bedienung von Experten voraussetzen. Lediglich für die Komponente der Präsentation von Geodaten, bestehen Möglichkeiten der Bedienung bzw. Nutzung durch eine breite Öffentlichkeit. Werden Geodaten in strukturierter Form über das Internet zur Verfügung gestellt, so spricht man von einem WebGIS (BILL 199b, 364 ff.).
Da die Präsentation dreidimensionaler Geometriedaten auf Basis gängiger WebGIS-Konzepte nicht möglich ist, beschränkt sich die Darstellung auf zweidimensionale Geometriedaten. Hierbei handelt es sich größtenteils um Karten, weswegen auch von WebMapping gesprochen wird. Die Nutzung solcher WebMapping-Dienste nimmt seit Jahren zu. Nach PETERSON (2003, 9 ff.) verdoppelt sich die Zahl mittels WebGIS generierter Karten von Jahr zu Jahr. Im Jahr 2001 wurden über das Internet bereits 21

Millionen Karten täglich abgerufen. Eine Einführung und Beispiele zum Thema Web-Mapping bzw. WebGIS liefern ASCHE & HERRMANN (2003).

Geographische Informationssysteme werden außer in der Geographie in den unterschiedlichsten Bereichen eingesetzt. Diese reichen von Archäologie und Bauingenieurwesen über Forstwirtschaft und Kartographie bis hin zu Raum- und Umweltplanung. Vorliegende Arbeit nutzt ein Geographisches Informationssystem zur Modellierung und Verwaltung von zweidimensionalen Geometriedaten aus den unterschiedlichsten Quellen. Mit der Konzeption und Implementierung des Informationssystems sollen dreidimensionale Geometriedaten sowie damit verknüpfte Sachdaten über das Internet verfügbar gemacht werden.

2.3.2 Kommunikation in Planungsprozessen

Wie in Kapitel 2.1.1 beschrieben sind einige Aspekte der Arbeit vor dem Hintergrund der Raumplanung zu sehen. Kommunikation in Planungsprozessen wird heute nicht mehr nur mit alten Verfahrensvorschriften in Planungsgesetzen, wie zum Beispiel Bürgerbeteiligung gemäß § 3 Baugesetzbuch (BauGB), gleichgesetzt. Kommunikation wird inzwischen im gesamtplanerischen Kontext gesehen. Sie spielt eine wesentliche Rolle bei der Konsensfindung im Rahmen baulicher Großmaßnahmen oder bei der Moderation gemeinschaftlicher Problemlösungen. Kommunikation in Planungsprozessen bedeutet Beschaffung und Austausch von Informationen. Die Beteiligung Dritter an Planungs- und Entwicklungsprozessen ist ebenso Teil dieser Kommunikation. Die Vielzahl der in den Planungsprozess involvierten Akteure hat spezifische Probleme bei der Kommunikation zwischen den einzelnen Parteien zur Folge. Die Kommunikation zwischen Fachleuten verschiedener Bereiche sowie zwischen diesen Fachleuten und Laien birgt typische Sender-Empfänger Konflikte, da nicht alle Beteiligten dieselbe Fachsprache sprechen. Kommunikation in Planungsprozessen beruht auf den drei Säulen von Information, Partizipation und Kooperation (BISCHOF ET AL. 2005, 10 ff.). Einer umfassenden Abhandlung der nachhaltigen Kommunikation im Zusammenhang mit Planungsprozessen widmet sich SELLE (2000).
Um in diesem weiten Feld Anknüpfungspunkte zu definieren, greift die Dissertation Aspekte konkreter Planungsebenen auf. Dies sind die kommunale Planungsebene sowie die Objektplanungsebene. Tabelle 1 stellt bezogen auf Heidelberg die verschiedenen Ebenen der Raumplanung, deren Akteure sowie Programme bzw. Pläne dar.

Planungs-ebene	Akteure (formell)	Programm, Plan (formell)	Akteure (informell)
Bund	BM f. Raumordnung, Ministerkonferenz f. Raumordnung (MKRO)	Raumordnungspolitischer Orientierungsrahmen (ORA, 1993), Raumordnungspolitischer Handlungsrahmen (HARA, 1995)	<u>Fachakademien</u> Akademie für Raumforschung und Landesplanung; Deutsche Akademie für Städtebau und Landesplanung <u>Spitzenverbände</u> Deutscher Städtetag; Deutscher Landkreistag; Deutscher Städte- und Gemeindebund
Land	Oberste Landesplanungsbehörde (BW: Wirtschaftsministerium)	Landesentwicklungsprogramm, Landesentwicklungsplan	
Regierungs-bezirk/ Kreis	Obere Landesplanungsbehörde (BW: Verband Region Rhein Neckar seit 01.01.06)	Regionalplan	
Kommune	Nachbarschaftsverband Heidelberg-Mannheim (18 Städte und Gemeinden)	Bauleitplanung: Flächennutzungsplan (FNP 2015/20 seit 15.07.06 rechtskräftig)	
	Baudezernat	Bauleitplanung: Bebauungsplan (informell: Rahmenplan)	
Objekt	Bauherr (Universitätsbauamt), Architekt	div. Pläne, Architekturmodelle	

Tab. 1: Übersicht der verschiedenen Raumplanungsebenen (eigene Darstellung)

Auf der kommunalen Ebene soll das Informationssystem Möglichkeiten zur Unterstützung der Bauleitplanung bieten. Hier insbesondere im Bereich der informativen Einbindung von Bürgern in den Planungsprozess. Diese so genannte Beteiligung der Öffentlichkeit am Planungsprozess ist in § 3 BauGB vorgeschrieben. Dort ist auch festgelegt, in welcher Form diese Beteiligung zu erfolgen hat. Bis vor einigen Jahren geschah dies mithilfe herkömmlicher, analoger Medien u.a. durch die öffentliche Auslegung der Entwürfe der Bauleitpläne für die Dauer eines Monats (KIRCHHOF & SCHMIDT-AßMANN 1990, 90/4). Mit dem Gesetz zur Anpassung des Baugesetzbuchs an EU-Richtlinien von 2004 hat das Baugesetzbuch umfangreiche Änderungen erfahren (BUNDESANZEIGER 2004, 1359 ff.). In § 4a erlaubt der Gesetzgeber den Kommunen erstmals den ergänzenden Einsatz von „elektronischen Informationstechnologien“ sowie des Internets. Die Absicht des Gesetzgebers ist die zeitliche Optimierung und verfahrenstechnische Vereinfachung der Öffentlichkeits- und Behördenbeteiligung (STEINEBACH & ALLIN 2005, 443). An der Form der Inhalte, u.a. dem Entwurf des Bauleitplanes, hat dies nichts geändert, lediglich die Medien zur Verbreitung der Inhalte sind moderner geworden. Umfangreiche Softwareprodukte decken inzwischen die formellen Aspekte des Beteiligungsverfahrens ab. Sie bieten Möglichkeiten zur Bereit-

stellung der Planungsunterlagen im Internet und offerieren dem Bürger auch die Möglichkeit, Einwände online zu äußern.
Der Einsatz solcher modernen Informations- und Kommunikationstechnologien auf Basis elektronischer Medien zum Zweck der öffentlichen Planung und Verwaltung ist unter dem Begriff E-Government bekannt. Die so genannte E-Partizipation wird heute als wesentlicher Bestandteil dieses Konzeptes gesehen (LÜHRS 2006, 130).
Neben den offiziell definierten Instrumenten zur Beteiligung der Öffentlichkeit existiert eine ganze Reihe informeller Möglichkeiten, wie die Öffentlichkeit einbezogen werden kann. Hier soll das Informationssystem ansetzen und durch die dreidimensionale Visualisierung geplanter Bauten einen gemeinsamen Zeichenvorrat bei Sender und Empfänger schaffen, den Fachleute wie Laien gleichermaßen zum Transport von Informationen nutzen können.

Auf der Ebene der Objektplanung soll das Informationssystem den Bauherren und Architekten Funktionen zur Verfügung stellen, die als Konzeptions- und Argumentationshilfe bei Vorplanungen oder bei Ausschreibungen und Wettbewerben geeignet sind. Hierzu wird das Konzept analoger Architekturmodelle aufgegriffen und Teile daraus digital abgebildet. Die Bedeutung von Architekturmodellen als Entwurfshilfe und Repräsentationsmittel ist bei BECKER (2002) dargelegt.

Vor dem beschriebenen planungswissenschaftlichen Hintergrund möchte die Dissertation mit dem dreidimensionalen Informationssystem eine Plattform schaffen, die die Konsensbildung und Entscheidungsfindung in Planungsprozessen auf verschiedenen Ebenen und unter Einbezug aller beteiligten Akteure unterstützt.

2.3.3 Dreidimensionale Stadtmodelle

Seit einigen Jahren arbeiten die Vermessungsämter großer Kommunen am Aufbau digitaler dreidimensionaler Stadtmodelle. In vielen Großstädten wie beispielsweise in Hamburg oder in Wien werden solche Modelle schon im Rahmen der Stadtplanung amtsintern genutzt (CIESLIK 2003, 254 ff.; DORFFNER & ZÖCHLING 2004, 90 ff.). Städte wie Berlin oder Essen setzen solche 3D-Modelle in der Wirtschaftsförderung ein, indem sie mithilfe der Modelle Standortvorzüge aufzeigen (BERLINER SENATSVERWALTUNG FÜR WIRTSCHAFT, ARBEIT UND FRAUEN 2006; BRENNER & KOLBE 2005, 106 ff.). Die Verfügbarkeit der 3D-Stadtmodelle ist dabei an die Infrastruktur kommunaler Informationssysteme gebunden. Dies beschränkt die Nutzung auf Amtsebene und macht zudem eine Bedienung durch Fachpersonal notwendig (BAUER & MOHL 2005, 265 ff.; KERSCHNER 2005, 336 ff.).

Die Form, in der die 3D-Stadtmodelle vorliegen, kann sich von Kommune zu Kommune enorm unterscheiden. Ein Unterschied ist der Detaillierungsgrad. Flächendeckend für das gesamte Stadtgebiet liegt üblicherweise nur ein sehr geringer Detailgrad des Modells vor. Meist handelt es sich dabei um Kubaturen, einfache Klötzchenmodelle, die den Grundriss einzelner Gebäude bzw. ganzer Baublöcke abbilden. Nur wenige Kommunen wie beispielsweise seit kurzem Berlin können mit detaillierteren 3D-Modellen für Teile ihres Stadtgebiets aufwarten. Diese integrieren neben der Dachform auch Fotos der Gebäudefassaden (SENATSVERWALTUNG FÜR STADTENTWICKLUNG BERLIN 2006). Auch die technischen Methoden zur Erhebung und Verwaltung der Daten weichen erheblich voneinander ab. Für die Erfassung dreidimensionaler Geodaten stehen verschiedene Methoden bereit. Hier werden tachymetrische, photogrammetrische und laserscanbasierte Methoden unterschieden. Erstere arbeitet mit der manuellen Einmessung einzelner Lagepunkte. Dies liefert zwar sehr genaue Daten, ist aber zur flächendeckenden Erfassung wenig effizient. Bei den photogrammetrischen Methoden werden aus Stereoluftbildern mit manuellen und halbautomatischen Verfahren einzelne Gebäude extrahiert (GÜLCH 2005, 4 ff.). Laserscan-Befliegungen liefern gewaltige Datenmengen, da hier spezifische Gebäudeteile wie beispielsweise Dachkanten nicht gezielt aufgenommen werden, sondern die gesamte Erdoberfläche als Punktwolke in einem bestimmten Rastermaß erfasst wird. Hierbei ist eine weitere Bearbeitung der Daten notwendig, bei der aus dem Oberflächenmodell einzelne 3D-Gebäude abgeleitet

werden (HAALA 2005, 26 ff.). Je nach eingesetzter Methode müssen die erfassten 3D-Daten mit bereits existierenden zweidimensionalen Daten, beispielsweise der Automatisierten Liegenschaftskarte der Vermessungs- und Katasterverwaltung, abgeglichen werden (DEUTSCHER STÄDTETAG 2005, 31). Neben den beschriebenen Methoden existieren weitere Ansätze, bei denen direkt auf vorhandenen 2D-Daten aufgebaut wird. Einer breiten Verwendung für verschiedenste Zwecke auch außerhalb der öffentlichen Verwaltung steht derzeit unter anderem die mangelnde Interoperabilität der 3D-Daten entgegen. Die verschiedenen Erhebungsmethoden implizieren unterschiedliche Datenmodelle, die darüber hinaus teilweise proprietäre, an bestimmte kommerzielle Hersteller gebundene Datenformate nutzen. Bestrebung ist es, ein standardisiertes Datenmodell zu entwickeln, das den Austausch von 3D-Stadtmodellen zwischen unterschiedlichen Verwaltungsplattformen erlaubt und eine Integrierung unterschiedlicher 3D-Modelle möglich macht (GRÖGER & KOLBE 2005, 234 ff.). Ein möglicher Kandidat für ein solch standardisiertes Datenmodell ist CityGML, dessen Entwicklung von der Special Interest Group 3D der Initiative Geodaten Infrastruktur NRW sowie dem Open Geospatial Consortium voran getrieben wird (OGC 2006). Ein aktuelles Forschungsprojekt am Geographischen Institut der Universität Heidelberg hat die Evaluierung verschiedener Erhebungsmethoden dreidimensionaler Geodaten zur Erstellung von 3D-Stadtmodellen zum Ziel. Ein Vergleich der Herangehensweisen großer deutscher Kommunen soll die unterschiedlichen Ansätze auf ein mögliches Best-Practice Verfahren hin untersuchen (GÖBEL 2007).

Die Zielgruppen und Anwendungsmöglichkeiten für 3D-Stadtmodelle sind vielfältig. Neben dem angesprochenen Einsatz in der öffentlichen Verwaltung für Lärmschutzkataster, Schadstoffmonitoring oder Stadtplanung sollen 3D-Stadtmodelle auch Eingang in mobile Navigationssysteme finden (GENSTORFER 2005; STRASSENBURG-KLECIAK 2006). Weitere Bereiche sind Wirtschafts- und Tourismusförderung, Funknetzplanung oder Immobilienbranche (BACHMANN ET AL. 2003; MÜLLER 2005; 392 ff.). Je nach Anwendungszweck stehen dabei die Visualisierung des 3D-Modells oder die Analyse im Vordergrund.

Vorliegende Arbeit greift einen bislang wenig beschriebenen Ansatz zur Erstellung von 3D-Stadtmodellen auf, um den Heidelberger Universitätscampus als dreidimensionales Modell digital abzubilden. Die gewählte Herangehensweise zieht dazu bestehende zweidimensionale Daten aus den verschiedensten Quellen heran, integriert diese

in ein GIS und bereitet sie dort für die weitere Verarbeitung auf. Im Gegensatz zu den oben angesprochenen, häufig eingesetzten photogrammetrischen und laserscanbasierten Verfahren fällt so zum einen der Aufwand zur Neuerhebung von Daten relativ gering aus und zum anderen kann eine Konsistenz zwischen bestehenden 2D-Daten und den neu generierten 3D-Daten gewährleistet werden. Für gewisse Teile des Modells wird zudem die Zeit als vierte Dimension abgebildet, um den Zustand bzw. einen möglichen Zustand zu bestimmten Zeitpunkten abbilden zu können.

2.3.4 3D-Visualisierung und 3D-Modellierung

Unter Visualisierung versteht man die Präsentation von Sachverhalten mittels visueller Medien. Die 3D-Visualisierung ermöglicht durch entsprechende Darstellung einen Einblick in dreidimensionale Daten; beispielsweise 3D-Geodaten in Form dreidimensionaler Stadtmodelle. Während analoge 3D-Modelle wie klassische Architekturmodelle aus Holz oder Kunststoff immer an einen zuvor bestimmten Maßstab gebunden und in ihrem Detailgrad beschränkt sind sowie selten in ihrem spezifischen Umfeld betrachtet werden können, bietet die computergestützte 3D-Visualisierung Möglichkeiten, die weit darüber hinaus gehen (MACH 2000, 26).
Die Art der Darstellung dreidimensionaler Daten kann variieren. Neben einfachen 3D-Animationen, die in Form eines Filmes Daten in einer für den Betrachter nicht beeinflussbaren Weise visualisieren, existieren immersive Umgebungen, die dem Publikum ein Eintauchen in die 3D-Welt und ein Interagieren mit den dreidimensionalen Inhalten erlauben. Die realitätsnahe Darstellung dreidimensionaler Objekte aus der Realwelt, die dem Benutzer eine Interaktion mit den Inhalten erlaubt, wird als Virtuelle Realität bezeichnet (POMASKA 2007, 28). Das Konzept der Virtuellen Realität kann dabei helfen, komplexe räumliche Informationen anschaulicher und nachvollziehbarer zu gestalten (BRODLIE ET AL. 2002, 9 ff.). Forschungen belegen, dass dreidimensionale virtuelle Modelle im Vergleich zu normalen zweidimensionalen Daten wie beispielsweise Karten wesentlich besser geeignet sind, um komplizierte räumliche Sachverhalte für Laien verständlich zu machen (CHEN & KNAPP 2006, 275).
Die 3D-Visualisierung von Daten über das Internet gewinnt zunehmend an Bedeutung, stellt dabei aber besondere Anforderungen an die 3D-Modelle und an die Plattform, welche Visualisierungs- und Interaktionskomponenten bereitstellt (ZACH ET AL. 2005, 210).

Die Grundlagen der 3D-Visualisierung liegen in der 3D-Computergrafik. Es existieren verschiedene Ansätze, um die Geometrie dreidimensionaler Objekte zu definieren bzw. zu repräsentieren. Das häufigst angewandte Verfahren ist die Polygon-Darstellung. Hierbei wird die geometrische Oberfläche eines Objektes durch ein Netz zusammenhängender Polygone, meistens Dreiecke, repräsentiert (MÜLLER 2005, 184). Die Beschreibung der räumlichen Lage der Polygone wird mittels Koordinaten vorgenommen, denen ein bestimmtes Raumbezugssystem zugrunde liegt. Meist werden einzelne 3D-Objekte zunächst in einem lokalen Koordinatensystem modelliert und später über eine Transformation in ein Weltkoordinatensystem eingefügt. Letzteres kann sich an geodätischen Referenzsystemen orientieren. Mit der Polygon-Darstellung lassen sich auch äußerst komplexe 3D-Objekte modellieren. Arbeitsaufwand und Hardware setzen jedoch in der Praxis Grenzen. Um ein 3D-Modell detaillierter erscheinen zu lassen, wird deshalb das Konzept des Texture-Mappings angewandt (POMASKA 2007, 54 ff.). Hierbei werden bildhafte Informationen, beispielsweise Rasterbilder von Gebäudefassaden, auf die Oberflächengeometrie von 3D-Modellen aufgebracht. Die Bildinformationen werden Texturen genannt. Der Prozess des Aufbringens ist auch als Texturierung oder Mapping geläufig.
Umfassenden Abhandlungen zum Thema Computergrafik bzw. Mapping widmen sich BUNGARTZ ET AL. (2002) sowie WATT (2000) bzw. EBERT ET AL. (2003).

Der Heidelberger Universitätscampus soll als texturiertes Polygonmodell die inhaltliche Grundlage des Informationssystems bilden. Das System selbst soll dem Nutzer in Form einer Virtual-Reality-Umgebung bestimmte Interaktionsmöglichkeiten mit den dreidimensionalen Inhalten erlauben. Das System soll die 3D-Visualisierung über das Internet bereitstellen.

3 Entwicklung des Untersuchungsgebiets

Seit der Gründung der Universität Heidelberg im Jahre 1386 ist ihr bauliches Abbild einem steten raumzeitlichen Wandel unterworfen gewesen. Dieser ist geprägt von einer räumlichen Expansion, die zur Folge hatte, dass sich die baulichen Ressourcen der Universität heute an verschiedenen Stellen des Stadtgebiets befinden.
Dieses Kapitel umreißt die Eckpunkte der Entstehung und baulichen Entwicklung des Universitätsgebiets „Im Neuenheimer Feld“. Eine detaillierte Abhandlung ist bei FREIWALD (2003) zu finden.

3.1 Expansionsphasen der Universität

Zu Beginn waren die baulichen Ressourcen der Universität Heidelberg durch eine verstreute Lage in der Kernaltstadt gekennzeichnet. Die Räumlichkeiten zur Unterbringung der Lehrer sowie zum Abhalten der Vorlesungen wurden in Bürgerhäusern und Klöstern angemietet. Nach Vertreibung der in Heidelberg sesshaften Juden durch den Pfalzgrafen Ruprecht I. bezog die Universität den Grundbesitz der ehemaligen jüdischen Gemeinde westlich der Heilig-Geist-Kirche (WOLGAST 1986, 6 ff.). Die universitären Einrichtungen waren ungleichmäßig in diesem ehemaligen Ghetto verteilt. In den nächsten Jahrhunderten nahmen die Studentenzahlen stetig zu. Während des Dreißigjährigen Krieges und des Pfälzischen Erbfolgekrieges erlitt die Entfaltung der Universität Rückschläge zum einen durch den Verlust ihrer Bibliothek und kurz darauf durch ihre Schließung. Als die Residenz 1720 nach Mannheim verlegt wurde, drohte die Universität in völliger Bedeutungslosigkeit zu versinken (WOLGAST 1986 55 ff.). Mit diesen äußeren Widrigkeiten kam der Bedarf auf, die Räumlichkeiten der Universität in einem Hauptgebäude unterzubringen. Anfang des 18. Jahrhunderts realisierte man dies durch den Bau der „Alten Universität“. Diese nahm Aula, Hörsäle, Bibliothek und wissenschaftliche Sammlungen auf. Der für den Lehr- und Forschungsbetrieb nachteiligen Streulage der Universitätsgebäude innerhalb des ehemaligen jüdischen Ghettos wollte man durch die räumliche Konzentrierung der Einrichtungen begegnen

(RÜCKBROD 1969, 31 ff.; RÜCKBROD 1977, 111 ff.). Im Jahr 1803 fiel Heidelberg an das Großherzogtum Baden. Dies hatte die Reorganisation der Universitätsstruktur zur Folge. Mit der finanziellen Absicherung und einer großzügigen Berufungspolitik begann der Aufstieg Heidelbergs zu einer der führenden Hochschulen Deutschlands (NÄGELKE 2000, 355 ff.). Neugründungen von Instituten folgten. Die Zahl der Professoren stieg von 77 auf mehr als das Doppelte. Im Jahr 1890 wurde schließlich eine weitere Fakultät, die Naturwissenschaftlich-Mathematische, eingerichtet (SCHWARZ 1996, 21 ff.).

Mit dem Aufstieg der Universität, den wachsenden Studentenzahlen und Institutsneugründungen war eine Expansion der Universitätsbauten nach Westen verbunden. 1803 verfügte die Universität lediglich über die 1735 fertig gestellte „Alte Universität“ sowie eine ihr zugewiesene Kameralsschule. Die Erweiterung nach Westen begann mit der Entscheidung, das ehemalige Dominikanerkloster in der westlichen Vorstadt zu kaufen. Dort sollten Kliniken, chemische Labors und das Anatomische Institut untergebracht werden (NÄGELKE 2000, 355 ff.).

Innerhalb der bestehenden Stadtgrenzen stand der Universität jedoch bald nicht mehr genügend Raum für eine Ausweitung des Wissenschaftsbetriebes zur Verfügung. Der hochschulbauliche Fokus verlagerte sich ebenso wie der städtebauliche Schwerpunkt immer weiter aus der Kernaltstadt hinaus. 1869 wurde das Klinikum im westlich gelegenen Stadtteil Bergheim gegründet. Dies war der dritte Standort der Universität. Bis Anfang des 20. Jahrhunderts entstand in Bergheim ein eigenes Klinikviertel aus rund 50 Gebäuden. Nach dem ersten Weltkrieg war die Zahl der Studenten auf mehr als 2600 angestiegen (KRÄMER 1986, 10 ff.).

Da zu Beginn des 20. Jahrhunderts viele Gebäude der Universität nicht den Anforderungen der Zeit entsprachen, waren entsprechende Renovierungen bzw. Neubauten notwendig. Erstmals forderte man statt bloßer Einzelmaßnahmen einen Generalbebauungsplan. Es kam der Gedanke auf, Naturwissenschaften und Medizin auf die nördliche Neckarseite zu verlegen (WOLGAST 1986, 123). In den folgenden Jahrzehnten wurde diese Expansion der Universität nach Norden Wirklichkeit.

Auf Verlangen des Kultusministeriums kaufte das Land Baden 1911 ein 20 Hektar großes Gelände westlich der Gemarkung Neuenheim. Dieses Neuenheimer Feld sollte dem Ausbau der Universität dienen. Die Erstellung eines ersten Generalbebauungsplanes wurde in Auftrag gegeben. Dieser lag ein Jahr später vor und umfasste zwei Varianten. Die erste Möglichkeit sah die stufenweise Verlegung aller Kliniken ins Neuenheimer Feld vor. Frei werdende Flächen auf Bergheimer Seite sollten von den Natur-

wissenschaften genutzt werden. Diese Alternative setzte jedoch den Bau einer dritten Neckarbrücke voraus. Da schon im Vorfeld absehbar war, dass eine solche radikale Umstrukturierung finanziell nicht zu bewerkstelligen sein würde, sah die zweite Alternative weniger radikale Schritte vor. Hiernach sollten der Botanische Garten und die Psychiatrische Klinik ins Neuenheimer Feld umziehen. Darüber hinaus sollten Neubauten für Zoologie, Mineralogie-Geologie, Chemie und Anatomie entstehen. Die Realisierung dieses Generalbebauungsplanes wurde durch den Ersten Weltkrieg vereitelt. Lediglich der Botanische Garten wurde 1913 als erste universitäre Einrichtung in das Neuenheimer Feld verlegt (KRÄMER 1986, 19 ff.).

3.2 Gesamtplanungen für das Neuenheimer Feld

Seit dem ersten Generalbebauungsplan von 1912 gab es bis heute eine ganze Reihe verschiedener Gesamtplanungen, die sich an den jeweils geltenden Umständen orientierten. Da sich die Planung dynamisch an verändernde Umstände anpasste, wurden jeweils lediglich Teile dieser Gesamtplanungen realisiert. Abbildung 1 veranschaulicht, welche Bereiche des Campus zu welcher Zeit gebaut wurden.

3.2.1 Der Generalbebauungsplan von 1932

Im Jahr 1926, als die desolaten Zustände, die der Erste Weltkrieg hinterlassen hatte, weitgehend beseitigt waren, hatte das Ministerium für Kultus und Unterricht ein neues Programm für die künftige bauliche Entwicklung der Universität Heidelberg angeregt. 1927 legte der Leiter des Badischen Bezirksbauamtes, Ludwig Schmieder, eine Planung für die Verlegung der Naturwissenschaften in das Neuenheimer Feld vor. Diese wurde nach Diskussion im Landtag für beschlussreif erklärt. Daraufhin errichtete die Stadt die Ernst-Walz-Brücke als dritte Neckarquerung zur Erschließung des Neuenheimer Feldes. Wegen der fortschreitenden wirtschaftlichen Depression konnten die von Schmieder projektierten Bauten für die Naturwissenschaften allerdings nicht mehr ausgeführt werden (KRÄMER 1986, 25 ff.). Obwohl die Realisierung der universitären Gebäude bis auf weiteres verschoben wurde, entstanden als Folge der neu errichteten Brücke die ersten Bauten für die Wissenschaften im Neuenheimer Feld. Im Jahr 1929 wurde das Kaiser-Wilhelm-Institut, das heutige Max-Planck-Institut, in unmittelbarer Nähe zum Brückenkopf der Ernst-Walz-Brücke fertig gestellt (LURZ & VOGT 1990, 94).

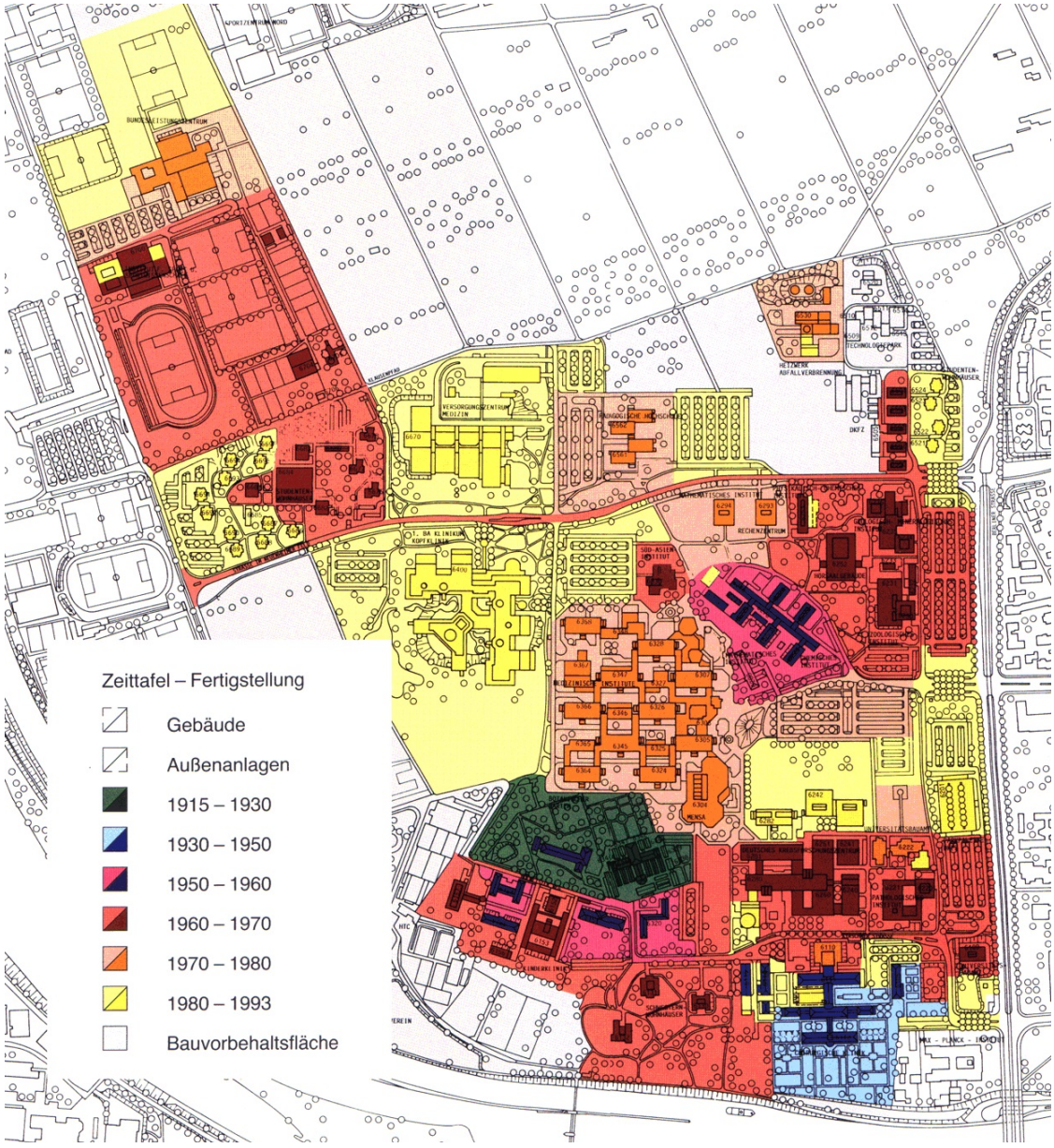

Abb. 1: Bauliche Entwicklung des Universitätscampus 'Im Neuenheimer Feld' (STAATLICHE HOCHBAUVERWALTUNG BADEN-WÜRTTEMBERG 1994, 9)

Da die Gebäude und Einrichtungen des alten Klinikgebiets in der westlichen Vorstadt inzwischen hoffnungslos veraltet waren und zudem in diesem Bereich keine Erweiterungsmöglichkeiten mehr bestanden, erteilte das Ministerium für Kultus und Unterricht Ludwig Schmieder den Auftrag für einen Generalbebauungsplan des Neuenheimer Feldes. Dieser sollte einen Neubau für das gesamte Klinikum und die Naturwis-

senschaften vorsehen (OVERBECK 1963, 101). Der als Schmiederplan bekannte Generalbebauungsplan lag 1932 vor. Die Planung sah vor, sämtliche Kliniken, also Chirurgische Klinik, Frauenklinik, Medizinische Klinik und Hals-Nasen-Ohrenklinik, in einem Band parallel zum Neckar anzuordnen (Abb. 2). Diese Lage sollte Lärmbelästigungen minimieren und durch die Südexposition die Genesung der Kranken beschleunigen. Schmieder entschied sich bewusst für eine solche weitgehend aufgelockerte Bebauung, in der jede Klinik ein eigenes Areal besitzen sollte. Gegenüber dem zu dieser Zeit häufig angewandten Konzept einer großen zentralen Bauanlage, sollte die vorgesehene Art der Bebauung die Nähe zu Gartenanlagen begünstigen und vermeiden, dass sich Patienten mit bestimmten Krankheiten räumlich zu nahe kommen. Dieser Generalbebauungsplan sollte die einheitliche Bebauung des Neuenheimer Feldes auch für kommende Zeiten sicherstellen. Deshalb umfasste die Planung nicht nur einen Flächenplan, sondern auch eine detaillierte räumliche Gestaltung der Gebäude, also Aufbau, Gebäudehöhen und Dachformen. Schmieder selbst schätzte den Zeitraum für die Realisierung dieser Gesamtplanung auf gut 100 Jahre (SCHMIEDER, 1938, 250 ff.).
Wie schon häufig stellte die Finanzierung der projektierten Bauten ein Hindernis dar. Mit der Machtübernahme der Nationalsozialisten 1933 erhielt das Klinikbauprojekt jedoch neuen Auftrieb, so dass zumindest die Finanzierung des Baus der Chirurgischen Klinik gesichert war. 1939 wurde die Klinik als einziger der projektierten Bauten fertig gestellt (Abb. 3) und konnte noch vor Ausbruch des Krieges bezogen werden (GRIESBACH & MAISANT 1986, 503 ff.).
Die im Schmiederplan projektierten baulichen Anlagen müssen hinsichtlich Umfang und Gestaltung auch vor dem Hintergrund des aufsteigenden Nationalsozialismus gesehen werden. Städtebau und Architektur spielten während dieser Zeit eine besondere Rolle (DÜLFFER ET AL. 1978). Sie dienten als Mittel zur Demonstration von Macht und Herrschaft. Die Errichtung der Chirurgischen Klinik wurde als nationalsozialistisches Prestigeobjekt behandelt (GRIESBACH & MAISANT 1986, 507). Der als fanatischer Ideologe geltende Minister des Kultus und Unterrichts Dr. Otto Wacker lobt die Schmiedersche Planung als „Möglichkeit, die Bauweise des 19. Jahrhunderts einmal grundsätzlich abzulösen und dem ganzen Erneuerungswerk den Stempel der neuen Zeit und neuen großräumigen Denkens zu geben“ (LANDESARCHIV BADEN WÜRTTEMBERG 2007; WACKER 1938, 5). Der Planung liege ein Leitgedanke zu Grunde, „der seine Kraft auch über größere Zeiträume zu erstrecken vermag“ (WACKER 1938, 6). Einer kritischen Auseinandersetzung mit dem Thema Architektur im Natio-

nalsozialismus unter besonderer Berücksichtigung Heidelbergs widmet sich FLECHTNER (2000).

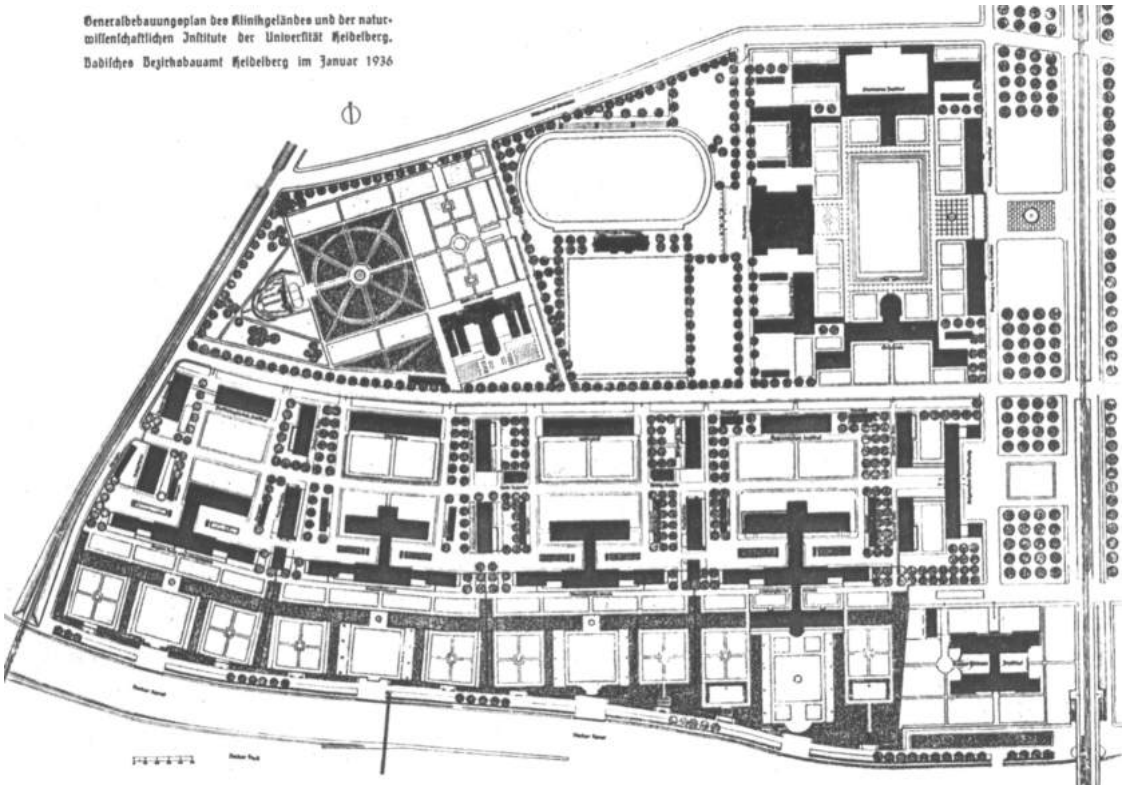

Abb. 2: Generalbebauungsplan 1932/33; ´Schmiederplan´ (SCHMIEDER 1936, 1)

Die Besonderheit des Schmiederplans gegenüber allen anderen Gesamtplanungen besteht darin, dass neben einer konzeptionellen Flächenplanung der gesamte Baukomplex bis hin zur Strukturierung von Fassaden detailliert beschrieben wurde. Die geschah in der Absicht, die entworfene Planung vollständig zu realisieren. Seit den 1950er Jahren beschränken sich die Gesamtplanungen auf eine wesentlich abstraktere, flächenhafte Planung, die Baugruppen ein bestimmtes Maß an Räumlichkeiten zuweist, die Ausgestaltung dieser Baugruppen jedoch offen lässt.

3.2.2 Entwicklung in den 1950er, 60er und 70er Jahren

Heidelberg hatte das Glück, im Zweiten Weltkrieg weitgehend unzerstört zu bleiben. Dennoch herrschten bis Ende der 1940er Jahre desolate Zustände, die weitere Planungen für das Neuenheimer Feld verhinderten. 1949 wurde das so genannte Klinikbaubüro gegründet. Als Vorläufer des heutigen Universitätsbauamts sollte es den Bauunterhalt bestehender Universitätsgebäude und die Errichtung universitärer Neubauten im Neuenheimer Feld auf Basis einer Gesamtplanung vorantreiben. Der Name „Klinikbaubüro“ spiegelt wider, dass dabei zunächst eine Verlegung der Kliniken bzw. deren Neubau im Mittelpunkt standen. Die zur Verfügung stehende Fläche von anfangs 20 Hektar wuchs durch Zukauf stetig an. Die Gesamtplanung von 1950 sah vor, auf einer Fläche von 65 Hektar sämtliche Institute der Universität im Neuenheimer Feld unterzubringen. Charakteristisches Merkmal der Gesamtanlage sollte eine parkartige Gestaltung des Geländes sein (KÖLMEL 1952, 144 ff.).

Auf Basis der Gesamtplanung von 1950 wurden die Chemischen Institute, die Mathematik sowie ein Großteil der Gebäude der Kinderklinik gebaut. Der Flächenzukauf gestaltete sich zum Teil langwierig, da Grundbesitz aus Privathand erworben werden musste. Städtebauliche Aspekte spielten deshalb bei der Anordnung der Institute eine untergeordnete Rolle. Dies ist an der diagonalen Lage der Chemischen Institute (Abb. 3) sowie der engen Nachbarschaft der Mathematik abzulesen (GORMSEN 1981, 116).

Im Jahr 1956 traf die Universität eine der bis heute wichtigsten Entscheidungen für die Beziehungen zwischen Stadt und Universität. Der große Senat beschloss, dass die Geisteswissenschaften ihren Standort in der Altstadt behalten sollten. Da der Planung von 1950 das Ziel eines vollständigen Umzugs ins Neuenheimer Feld zu Grunde lag, wurden neue Planungen notwendig.

Im August 1957 wurde zu diesem Zweck das Universitätsbauamt gegründet. Fortan lag den Planungen ein gänzlich anderes Konzept zugrunde. Bei bisherigen Planungen war es nicht gelungen, die zukünftige Entwicklung des Raum- und Flächenbedarfs vorauszusehen. Da auch die Stadtplanung in dieser Zeit keine brauchbaren Ansätze besaß, war die Anordnung der bislang gebauten Universitätsgebäude trotz allem von einer gewissen Planlosigkeit gekennzeichnet (GORMSEN 1981, 116). Das Universitätsbauamt verfolgte nun ein Konzept, bei dem die Aufstellung einer Gesamtplanung ein dynamischer Prozess sein sollte. Dies bedeutete, dass die Gesamtplanung fortlaufend aktuali-

siert und angepasst wurde, um den sich im Laufe der Jahre verändernden Vorraussetzungen Rechnung zu tragen. Im Rahmen dieser dynamischen Planung wurden seit 1960 eine große Anzahl von Einzelbauten und Gebäudekomplexen im Neuenheimer Feld errichtet. Deren Abfolge wurde dabei von Dringlichkeit und Verfügbarkeit finanzieller Mittel bestimmt (SCHWARZ 1996, 22).

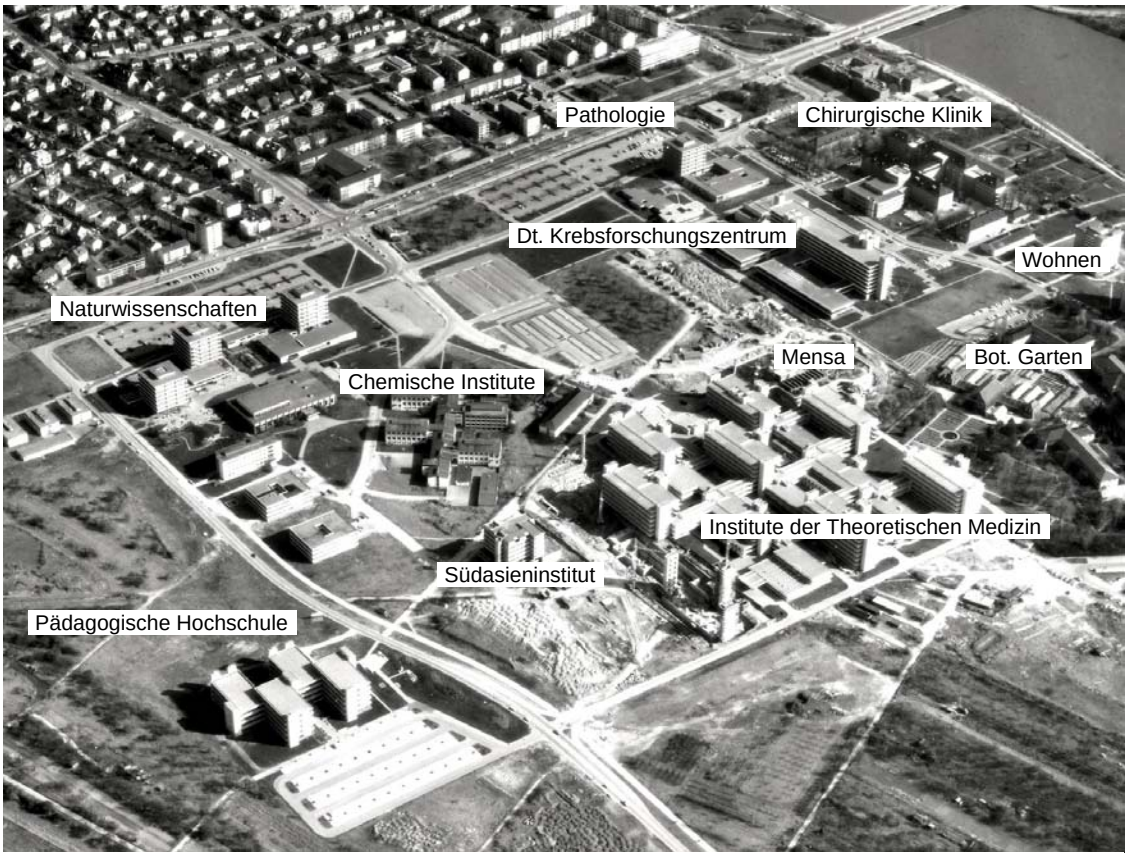

Abb. 3: Luftaufnahme des Neuenheimer Feldes von 1973, Blick aus Nordwesten (BILDARCHIV UNIVERSITÄTSBAUAMT HEIDELBERG)

Neben der Planung von Gebäuden spielte fortan auch die Entwicklung einer angepassten Verkehrsinfrastruktur zur Erschließung des Neuenheimer Feldes eine Rolle. Als maßgebliches Motiv sollte der Verkehr nur an das Neuenheimer Feld herangeführt werden bzw. es umfließen. So sollte eine geringst mögliche Lärmbelastung für Lehr- und Forschungstätigkeit wie auch für die Kliniken gewährleistet werden (WERKLE 1959, 56). Bereits in der Planung von 1960 kam dadurch dem am nördlichen Rand des beplanten Gebiets in West-Ost Richtung verlaufenden Klausenpfad eine besondere Position zu. In seiner westlichen Verlängerung war schon eine Querung des Neckars

angedacht (STAATLICHE HOCHBAUVERWALTUNG BADEN-WÜRTTEMBERG 1994, 7). Seit damals wurde eine Reihe grundlegender verkehrsplanerischer Elemente definiert, die in der jeweils aktuellen Gesamtplanung verfolgt wurden. Hierzu gehören die weitgehende Freihaltung des Universitätsgebiets vom Fahrverkehr. Das Parken sollte möglichst am Rande des Geländes, möglichst integriert in künftige Hochbauten stattfinden. Die innere Erschließung des Areals sollte durch zwei große, sich kreuzende Fußgängerbereiche erfolgen, die gleichzeitig die Bebauung in Nord-Süd- und West-Ost-Richtung gliedern sollten. Zentrale Einrichtungen der Universität sollten entlang dieser Bereiche angeordnet werden (UNIVERSITÄTSBAUAMT HEIDELBERG 2002, 3).

Auf der Grundlage konzeptioneller Planungen des Universitätsbauamts entstand 1961 ein erster amtlicher Bebauungsplan für das Neuenheimer Feld. Hierin wies das Planungsamt der Stadt Heidelberg für die Universität eine Vorbehaltsfläche von 120 Hektar aus. Mit der Festsetzung von Freiflächen- und Geschossflächenzahl wurde das Maß der baulichen Nutzung festgelegt. Weitere Restriktionen wie die Ausweisung überbaubarer Flächen, Straßenfluchten oder Gebäudehöhen wurden bewusst nicht vorgenommen, um die bauliche Entwicklung der Universität so flexibel wie möglich zu gestalten.

Wegen steigender Studentenzahlen stand der Hochschulbau Ende der 1950er Jahre unter höchstem Quantitätendruck. Als Folge erlebten die deutschen Hochschulen in den 1960er und 70er Jahren eine enorme bauliche Expansion. Die Bauweise war von starker Typisierung und Standardisierung geprägt. Es kristallisierte sich ein Bautypus heraus, der sich gleichzeitig durch ein Höchstmaß an Wirtschaftlichkeit und funktionaler Flexibilität unter Wegfall passiver oder dekorativer Elemente auszeichnete (KOCH 1991, 88). Bezogen auf Heidelberg spiegelt sich diese funktionsneutrale Typenbauweise in den zwischen 1960 und 1970 errichteten Gebäuden für die naturwissenschaftlichen Institute im Nordosten des Neuenheimer Feldes wider (Abb. 3). In diese Zeit fällt auch die Errichtung der ersten Studenten- und Schwesternwohnheime im Süden und Westen des Campus. Die ersten medizinischen Institute im Süden und Südosten des Universitätsgebiets wurden ebenfalls in dieser homogenen Einheitsbauweise errichtet (Abb. 3). Um diese monotonen Betonwüsten, die dem Nutzer nur wenig Orientierungsmöglichkeiten bot, besser in die Landschaft einzubinden, gab es seit den 1960er Jahren auch Grünplanungen zur Gestaltung der unbebauten Freiräume (SCHMITT 1986, 520 ff.). Der zentrale Bereich des Neuenheimer Feldes blieb nach wie vor einer Bebauung durch das Klinikum, die Institute der Theoretischen Medizin und der Mensa vorbehalten. Diese wurden 1975 in einer leicht modifizierten Typenbauweise auf Basis

der Gesamtplanung von 1972 errichtet. Diese Gesamtplanung wollte wegen der rapide gestiegenen Studentenzahlen eine maximal mögliche Verdichtung der Bebauung erreichen. Ausgenommen von dieser Verdichtung waren die Wohnbereiche am Neckar, der Botanische Garten sowie die Fußgängerbereiche und Grünzüge, die den Campus gliedern sollten. Mit der Fertigstellung von Mensa und Theoretischer Medizin zeichnete sich erstmals das den vergangenen Gesamtplanungen zu Grunde liegende Konzept eines Zentralbereichs räumlich ab. Als gliederndes Element zieht sich eine zentrale Achse von der Pädagogischen Hochschule im Norden über das Gebiet zwischen Südasieninstitut und Chemischen Instituten. Diese Achse erweitert sich im Bereich östlich der Theoretischen Medizin und Mensa platzartig und geht zwischen Wohnbauten und Chirurgischer Klinik schließlich in den Grüngürtel am Neckar über.

3.2.3 Entwicklung seit den 1980er Jahren bis heute

Mitte der 1970er Jahre waren mit den bislang gebauten Einrichtungen die Weichen für die weitere bauliche Entwicklung gestellt. Die Gebäude, die zur Zeit der größten universitären Expansion realisiert worden waren, gaben einen engen Rahmen für künftige Planungen vor. Der Standort des künftigen Klinikums war durch die Lage der theoretischen medizinischen Institute vorbestimmt. Die Lage des künftigen zentralen Bereichs des Neuenheimer Feldes war an die unmittelbare Nähe zur Mensa gebunden. Insgesamt waren die großen Baumaßnahmen mit Ausnahme des integrierten Klinikums abgeschlossen. Bereits in den 60ern hatte man erkannt, dass eine bauliche Realisierung des Klinikums, das sämtliche Universitätskliniken vereinen sollte, aufgrund der beschränkten Finanzierung aus Landesmitteln nur über mehrere Einzelbauabschnitte möglich wäre (SCHMITT 1986, 533 ff.).

Der Bau des ersten Abschnitts des Gesamtklinikums zog sich wegen immer wieder auftretender finanzieller Engpässe über neun Jahre hin. 1987 wurde die westlich der Theoretischen Medizin liegende Kopfklinik in Betrieb genommen. Sie vereint Mund-Zahn-Kiefer-Klinik, Augenklinik, Hals-Nasen-Ohren-Klinik, Neurologische Klinik, Neurochirurgische Klinik und Radiologische Klinik. Das Gelände südlich und westlich dieser neu errichteten Klinik wurde für die folgenden Bauabschnitte des Klinikums freigehalten. Die Gesamtplanung von 1994 sieht die Anordnung der weiteren Bauabschnitte des Klinikums in Form eines Ringes vor. Der zweite Bauabschnitt umfasst die Medizinische Klinik und wurde Ende 2003 fertig gestellt (UNIVERSITÄTSKLINIKUM HEIDELBERG PLANUNGSGRUPPE MEDIZIN 2005). Das Konzept des so genannten Hei-

delberger Klinikrings ist auch in der aktuellen Planung von 2004 vorgesehen. Dem Bauantrag für den ersten Bauabschnitt der Physikalischen Institute wurde 1997 statt gegeben. Der Bau wurde 2002 abgeschlossen. Weitere zwei Bauabschnitte sollen folgen.

3.3 Zukünftige Entwicklung

Langfristiges Ziel der baulichen Entwicklung des Neuenheimer Feldes ist die Integration des Universitätscampus in die städtebauliche Gesamtordnung. Neben weiteren Baumaßnahmen für Gebäude gehört hierzu die infrastrukturelle Ausgestaltung der Gesamtanlage. Beides soll sich an einem vorgegebenen städtebaulichen Funktions- und Nutzungsschema orientieren (Abb. 4).

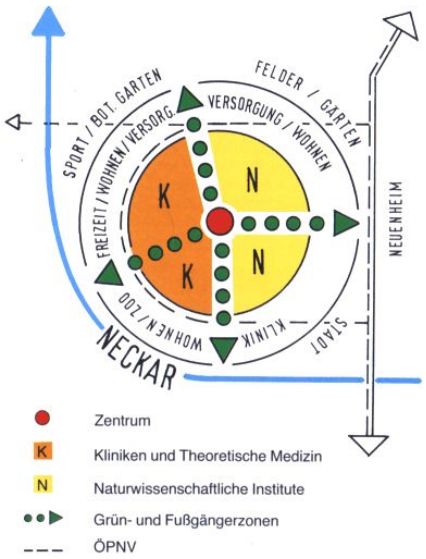

Abb. 4: Städtebauliches Funktions- und Nutzungsschema für den Universitätscampus (STAATLICHE HOCHBAUVERWALTUNG BADEN-WÜRTTEMBERG 1994, 14)

Abbildung 5 zeigt die derzeit aktuelle Gesamtplanung von 2004. Der Baubestand ist lilafarben dargestellt, während die Planung hellbraun hervorgehoben ist. Gestreifte Bereiche kennzeichnen Baufenster.

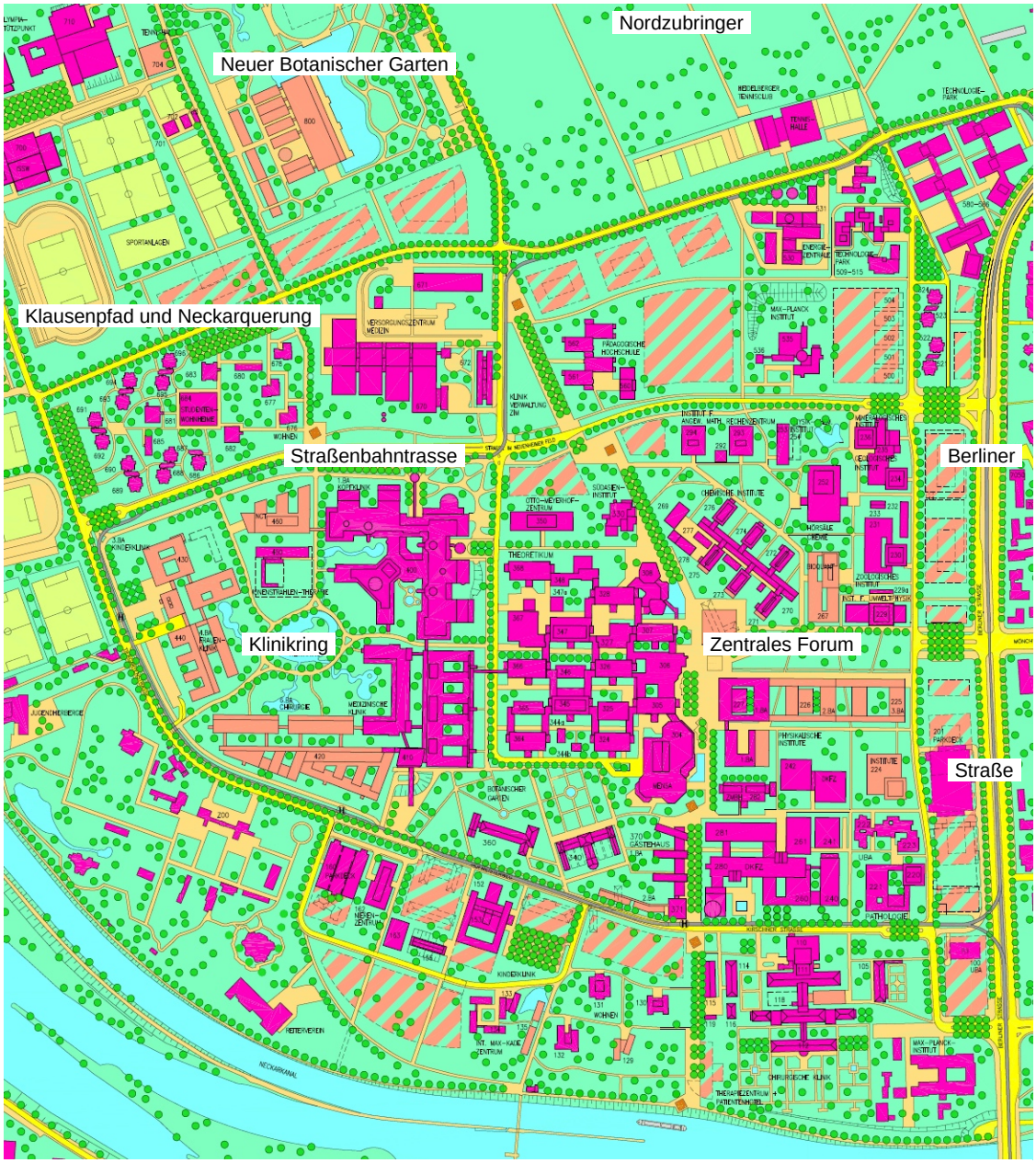

Abb. 5: Aktuelle Gesamtplanung für den Universitätscampus (UNIVERSITÄTSBAUAMT HEIDELBERG 2004b)

Wie schon in der Gesamtplanung 1994 vorgesehen und in der von 2004 fortgeführt, bilden zwei sich kreuzende, nord-süd bzw. ost-west verlaufende Fußgängerbereiche ein zentrales, gliederndes Element der Planung. Mit den an sie angrenzenden Grünflächen soll die flächenhafte Bebauung gegliedert und gefasst werden. Sie dienen zudem

als Verteiler für die fußläufige Erschließung des gesamten Areals. Der Schnittpunkt dieser beiden Achsen wird das bauliche Zentrum des Universitätscampus bilden. Dieses zentrale Forum (Abb. 5) soll weiter ausgebaut werden. Mittel- bis langfristig ist die Realisierung der weiteren Bauabschnitte des Physikalischen Instituts sowie die Errichtung eines Universitäts-Servicezentrums für Information und Kommunikation geplant (UNIVERSITÄTSBAUAMT HEIDELBERG 2002, 6ff.). Im Jahr 2007 soll der 2004 begonnene Bau des Bioquant-Gebäudes fertig gestellt werden, welcher wie auch die Chemischen Institute an den Forumsbereich angrenzt. Letztere sollen mittelfristig durch Neubauten an gleicher Stelle ersetzt werden. Die Achse, die vom Zentralbereich nach Norden führt, soll zukünftig die Verbindung zum nördlich des Klausenpfades projektierten neuen Botanischen Garten herstellen (Abb. 5). Nach Süden hin ist der Anschluss des zentralen Forums an das Neckarufer geplant. Eine Anlegestelle für eine regelmäßige Fährverbindung zum Altklinikum in Bergheim sowie zu den Instituten in der Altstadt steht als Idee im Raum. Die vom Zentralbereich nach Osten Richtung Berliner Straße führende Achse soll den neuen Hauptzugangsbereich des Campus bilden. Nach dem Vorbild amerikanischer Campusanlagen ist ein breiter Grünraum mit sich kreuzenden Wegen geplant. Sowohl für die Nord-Süd-Achse wie auch für die östliche Achse ist eine alleeartige Ausgestaltung vorgesehen. Nach Westen hin soll vom Forum ausgehend eine neue Achse geschaffen werden. Diese soll durch den Bereich der Institute für Theoretische Medizin führen und über den Patientengarten im Zentrum des Heidelberger Klinikrings mit den Grünanlagen und städtischen Freizeiteinrichtungen entlang des Neckars verbinden (MÜLLER 1998, 262; UNIVERSITÄTSBAUAMT HEIDELBERG 2002, 6ff.). Neben dem Ausbau des zentralen Forums und der inneren Erschließung des Universitätsgebiets steht mittel- bis langfristig die Realisierung der Planungen für den Heidelberger Klinikring an (UNIVERSITÄTSKLINIKUM HEIDELBERG PLANUNGSGRUPPE MEDIZIN 2005). Dieses Vorhaben steht seit dem ersten Generalbebauungsplan im Fokus jeder nachfolgenden Gesamtplanung.

Wesentliche Punkte, die der Verkehrsplanung für das Universitätsgebiet zugrunde liegen, wurden bereits in einer entsprechenden Untersuchung 1968 festgesetzt (UNIVERSITÄT HEIDELBERG 1968, 33 ff.). Die inneren Bereiche sollten demnach der fußläufigen Erschließung vorbehalten bleiben, während der motorisierte Verkehr nur an den Campus herangeführt werden sollte. Die Bereitstellung verkehrstechnischer Infrastruktur ist Aufgabe der Stadt. Laut einem Generalvertrag von 1962 hat sie für eine funktionsfähige äußere Erschließung zu sorgen. Hierfür ist eine finanzielle Unterstützung des Landes notwendig (FISCHER 2004, 57 ff.). Derzeit wird das Neuenheimer Feld allein über

die Berliner Straße im Osten erschlossen. Diese ist als vierspurige Hauptverkehrsstraße ausgebaut und schließt den Campus über vier Zufahrten an. Während der Stoßzeiten ist die Berliner Straße überlastet. Sie gilt als die am stärksten belastete innerörtliche Hauptverkehrsstraße in Heidelberg (STADT HEIDELBERG 2005a).

Seit Jahren, für einzelne Maßnahmenvorschläge sogar seit Jahrzehnten, wird darüber diskutiert, wie man die äußere Erschließung des Neuenheimer Feldes verbessern könnte. Die Vorschläge reichen dabei von Maßnahmen zur Begrenzung des Individualverkehrs wie Parkraumbewirtschaftung und Ausbau des öffentlichen Personennahverkehrs bis hin zu großen Lösungen, die auf einen direkten Anschluss des Campus an das überörtliche Straßennetz abzielen (BAUER & TEUFEL 2004, 4 ff.). Bereits 2004 wurde vom Gemeinderat eine durch das Feld führende Straßenbahntrasse beschlossen (STADT HEIDELBERG 2004). Verschiedene Varianten über den genauen Verlauf werden zur Zeit diskutiert. Für den Anschluss an das überörtliche Verkehrswegenetz standen und stehen verschiedene Maßnahmen zur Debatte. Die schon lange diskutierte Variante eines verkehrstechnischen Ausbaus des Klausenpfades (Abb. 5) inklusive einer westlichen Fortführung über den Neckar zum Autobahnanschluss Rittel hat mit einer vom Gemeinderat in Auftrag gegebenen Umweltverträglichkeitsuntersuchung 2005 einen Dämpfer erlitten. Der Bereich des Altneckars, den eine mögliche fünfte Neckarbrücke queren würde, ist ein europäisches Naturschutzgebiet (Fauna-Flora-Habitat), dessen Schutzwürdigkeit die Umweltverträglichkeitsuntersuchung bestätigt hat (STADT HEIDELBERG 2005b). Die Möglichkeit einer Untertunnelung des Neckars in diesem Bereich stellt Stadt und Land vor große finanzielle Probleme (BAUER & TEUFEL 2004, 4 ff.). Alternativ zur fünften Neckarquerung wird eine Verbindung des Neuenheimer Feldes mit dem Autobahnanschluss Dossenheim diskutiert. Diese ist unter dem Schlagwort „Nordzubringer“ (Abb. 5) bekannt. Für diesen sind wiederum mehrere Alternativen in Diskussion, die ebenfalls in der Umweltverträglichkeitsuntersuchung beleuchtet werden. Unabhängig von diesen Diskussionen gibt es konkretere Planungen, die Berliner Straße zu einem städtischen Boulevard auszubauen. Der Stadtteilrahmenplan für Neuenheim sieht vor, dass gemeinsam mit Universität und übrigen Anliegern ein entsprechendes Konzept ausgearbeitet werden soll, wie der Straßenraum auf beiden Seiten zukünftig gestaltet werden kann. Eine Nutzungsmischung ist dabei erklärtes Ziel (STADT HEIDELBERG 2002). Bereits 2005 wurden vom Gemeinderat Schritte zur Erstellung eines Baumassenkonzeptes für die Berliner Straße eingeleitet (STADT HEIDELBERG 2005). Nach Informationen des Stadtplanungsamts soll ein solches Konzept Ende 2007 vorliegen.

4 Konzeptionelle Zielsetzungen

Für die Realisierung der beiden Kernziele der Arbeit, nämlich die Schaffung eines digitalen, dreidimensionalen Abbilds des Untersuchungsgebiets sowie die Umsetzung eines interaktiven 3D-Informationssystems, ist es notwendig im Vorfeld spezifische Anforderungen zu postulieren.

4.1 3D-Modell

Das dreidimensionale virtuelle Computermodell des Heidelberger Universitätscampus sowie die Methoden zu dessen Erstellung sollen bestimmten Ansprüchen genügen. Die grundsätzlichen sowie technischen Anforderungen basieren auf Erfahrungen und Erkenntnissen, die der Verfasser während verschiedener 3D-Projekte in Forschung und Industrie sammeln konnte. Die inhaltlichen Anforderungen leiten sich aus dem Untersuchungsgebiet selbst sowie diesbezüglich bestehender zukünftiger bzw. historischer Planungen ab.

4.1.1 Grundsätzliche und technische Anforderungen

Nachfolgend wird definiert, wie das 3D-Modell beschaffen sein soll, welche spezifischen Eigenschaften es besitzen soll und welche Ansprüche an die Methoden zu seiner Erstellung gestellt werden:

- Eine möglichst universelle Nutzungsmöglichkeit des 3D-Modells soll gewährleistet sein: Das Modell soll gleichermaßen für fotorealistische Visualisierungen wie auch für interaktive Online-Anwendungen geeignet sein. Dies ist insofern eine Herausforderung als die gesetzten Anwendungsziele gegenläufige Prämissen haben. Ein 3D-Modell für die interaktive Anwendung im Internet einerseits sollte ein möglichst geringes Datenvolumen haben, um die Datenübertragung zeitlich in Grenzen zu halten sowie um das Berechnen und Darstellen des Modells auch auf gering performanter Hardware einzelner Internet-

nutzer in Echtzeit zu gewährleisten. Ein 3D-Modell für die Herstellung hochwertiger, quasifotorealistischer Filme oder Abbildungen andererseits muss möglichst detailliert sein, was das Datenvolumen massiv erhöht. Durch automatisierbare Veränderbarkeit gewisser Grundelemente des 3D-Modells soll ein Hybridmodell realisiert werden, welches diesen unterschiedlichen Ansprüchen gleichermaßen genügt.

- Das 3D-Modell soll sich konsistent zu bestehenden 2D-Daten verhalten: Ansätze, die zur Generierung von dreidimensionalen Stadtmodellen photogrammetrische oder laserscanbasierte Verfahren nutzen, machen im Anschluss einen zum Teil aufwendigen Abgleich mit bereits vorliegenden zweidimensionalen Daten wie beispielsweise dem Amtlichen Liegenschaftskataster notwendig. Um diesbezüglich eine Konsistenz zu gewährleisten, soll das 3D-Modell direkt auf den entsprechenden, bereits vorhandenen 2D-Daten aufbauen.
- Die Prozesskette für Erhebung, Verwaltung und Verarbeitung von Daten soll so gestaltet sein, dass Pfadabhängigkeiten vermieden werden: Das 3D-Modell soll auf bestehenden 2D-Daten verschiedener Quellen aufbauen. Diese müssen für die dreidimensionale Modellierung aufbereitet werden. Statt dies direkt und ausschließlich in der zur dreidimensionalen Modellierung ausgewählten Softwareumgebung zu bewerkstelligen, soll eine Prozesskette konzipiert werden, die es erlaubt die zweidimensionalen Daten verschiedenen Nutzungsmöglichkeiten zuzuführen. Hiermit wird gewährleistet, dass die Inhalte, die das 3D-Modell bzw. das 3D-Informationssystem bereit halten soll auch für andere Plattformen verfügbar gemacht werden können. Dies soll erreicht werden, indem die Daten zunächst in ein GIS integriert werden.
- Der Aufwand für die Neuerhebung von Daten soll so gering wie möglich ausfallen: Für die Erstellung von 3D-Modellen stehen unterschiedliche technische Methoden zur Verfügung. Die Methoden unterscheiden sich hinsichtlich des notwendigen finanziellen Aufwands zur Datenbeschaffung, der anschließenden Auswertung wie auch in ihrer geometrischen Genauigkeit des fertigen 3D-Modells. Für vorliegende Arbeit soll eine Methode gewählt werden, die soweit wie möglich vorhandene Daten nutzen kann und kostenintensive Neuerhebungen vermeidet.

- Das 3D-Modell soll in ein einheitliches räumliches Bezugssystem eingebettet sein: Eine solche Georeferenzierung erleichtert die Integration zusätzlicher 2D- und 3D-Daten in das Modell und unterstützt den Datenaustausch mit anderen georeferenzierten Modellen. Darüber hinaus erlaubt die Georeferenzierung raumbezogene Analysen innerhalb des Modells. Die oben genannte Integration in ein GIS ermöglicht die Einbettung der Daten in ein einheitliches Raumbezugssystem.
- Das 3D-Modell bzw. die Art seiner Generierung soll eine möglichst hohe Investitionssicherheit bieten: Bei der Modellierung sollen deshalb überwiegend Techniken und Werkzeuge zur Anwendung kommen, die als Standards oder Quasistandards im Bereich der 3D-Modellierung gelten. So kann sichergestellt werden, dass das Modell mit Standardwerkzeugen gepflegt und über eine lange Zeitspanne genutzt werden kann.
- Das 3D-Modell soll einen einheitlichen Charakter besitzen: Um einen homogenen und natürlichen visuellen Eindruck des 3D-Modells zu erreichen, welcher seine Verwendbarkeit und die Akzeptanz des Betrachters erhöht, sollen der Detaillierungsgrad und der Charakter einzelner Gebäude innerhalb des Modells möglichst wenig differieren. Der in diesem Rahmen angestrebte Einsatz von Gebäudefassaden als Texturen soll so erfolgen, dass die Fassadenfotos möglichst perspektivisch unverzerrt sind und keine störenden, nicht zur Fassade gehörenden Bildelemente enthalten. Dies macht eine aufwendige manuelle Bearbeitung der Fotos notwendig.

4.1.2 Inhaltliche Anforderungen

Nachfolgend wird definiert, welche Inhalte das 3D-Modell darstellen soll bzw. nach welchen Kriterien diese Inhalte ausgewählt wurden und warum deren Integration als zweckmäßig erachtet wird:

- Das Modell soll den aktuellen Baubestand vollständig abbilden: Erst mit einer flächendeckenden Abbildung der baulichen Ressourcen des Untersuchungsgebiets ist eine Verwendung für den in dieser Arbeit angestrebten Zweck eines 3D-Informationssystems sinnvoll.
- Die 3D-Modellierung soll die bestehende Verkehrsinfrastruktur vollständig abbilden: Straßen, Parkplätze, Fußwege sowie dazwischen liegende Grün- und

Wasserflächen sollen Teil des dreidimensionalen Modells sein. Neben den Grünflächen wird außerdem eine Abbildung des aktuellen Baumkatasters angestrebt. Die genannten Elemente sind wesentliches Merkmal des Untersuchungsgebiets. Die Visualisierung zukünftiger verkehrtechnischer Planungen ist ohne eine Darstellung des aktuellen Bestands nicht sinnvoll.

- Ein 3D-Geländemodell soll umgesetzt werden: Ebenso wie die Verkehrsinfrastruktur ist das Gelände ein prägendes Element. Markante Reliefunterschiede innerhalb des Untersuchungsgebiets sollen digital modelliert werden. Hierzu gehören großflächige, charakteristische Abgrabungen im Bereich der Kliniken. Zudem sollen das Neckarbett und der Neckarkanal auf einer Länge von rund drei Kilometern digitalisiert und in das Geländemodell integriert werden.
- Das Modell soll ausgewählte zukünftige Bauvorhaben dreidimensional abbilden: Die Auswahl basiert auf verschiedenen Kriterien. Zum einen müssen belastbare, das heißt verlässliche und aussagekräftige Daten für eine dreidimensionale Modellierung vorhanden sein. Zum anderen müssen die Vorhaben so geartet sein, dass sie das Untersuchungsgebiet in seinem baulichen Erscheinungsbild bzw. Charakter deutlich verändern werden. Auch die Rolle, die gewisse Vorhaben oder Pläne in der öffentlichen Diskussion spielen, wurde bei der Auswahl berücksichtigt. Um festzustellen, welche Vorhaben in Frage kommen, wurden formelle sowie informelle Planwerke wie die Gesamtplanung des Universitätsbauamts oder der Stadtteilrahmenplan für Neuenheim analysiert (vgl. Kap. 3.3). Darüber hinaus wurden mit den in diesem Bereich offiziell verantwortlichen Akteuren Gespräche geführt. Für eine Modellierung wurden die für den zentralen Bereich des Neuenheimer Feldes geplanten oder bereits konkret projektierten Gebäude ausgewählt (vgl. Kap. 3.3). Das zentrale Forum ist der bauliche Mittelpunkt und das am häufigsten frequentierte Areal des Universitätscampus. Es trägt damit maßgeblich zum zukünftigen Gesamteindruck des Gesamtareals bei. Konkret wurden hier folgende Vorhaben ausgewählt: die weiteren Bauabschnitte für die Physik, das Bioquantgebäude, die Neubauten für die Chemischen Institute sowie das geplante universitäre Servicezentrum für Information und Kommunikation. Im Fokus der aktuellen Gesamtplanung steht wie in allen vorangegangenen Entwürfen auch die vollständige Realisierung des integrierten Gesamtklinikums. Baulicher Umfang und finanzieller Rahmen zeichnen dieses Projekt als wichtigstes Vorhaben der Gesamtplanung

aus. Aus diesem Grund soll der so genannte Heidelberger Klinikring ebenfalls Inhalt der dreidimensionalen Modellierung sein. Im öffentlichen Dialog spielt die zukünftige verkehrstechnische Erschließung des Neuenheimer Feldes eine herausragende Rolle (vgl. Kap. 3.3), deshalb sollen die verschiedenen Varianten der zur Diskussion stehenden fünften Neckarquerung ebenso in das 3D-Modell einfließen wie die Alternativen des so genannten Nordzubringers durch das Handschuhsheimer Feld.

- Ausgewählte historische Bauvorhaben sollen als dreidimensionale Inhalte abgebildet werden: Da bauliche Planung anders als früher heute als ein dynamischer Prozess verstanden wird, der permanent an den aktuellen Bedarf und hinsichtlich der architektonischen Ausgestaltung an den jeweils herrschenden Zeitgeist angepasst wird, erscheint eine Auseinandersetzung mit der historischen Planung für das Untersuchungsgebiet reizvoll. Die Auswahl historischer Bauvorhaben basiert auf denselben Kriterien wie bei den Objekten der zukünftigen Planung. Hierzu wurden Generalbebauungspläne und Gesamtplanungen seit 1912 untersucht (vgl. Kap 3.2 bzw. FREIWALD 2003, 70 ff.). Die Verlegung bzw. der damit verbundene Neubau der Universitätskliniken im Neuenheimer Feld war bei allen Planungen von 1912 bis heute ein zentrales Thema. Auch der Generalbebauungsplan von 1932 stellt den Neubau eines Gesamtklinikums in den Mittelpunkt (Kap. 3.2.1). Er spielt aus zwei Gründen eine besondere Rolle. Zum einen zeichnet er sich im Vergleich zum heute geltenden Planungsverständnis durch einen absolut geprägten Charakter aus. Der Generalbebauungsplan war mit dem Ziel einer vollständigen Realisierung entworfen worden. Zum anderen sieht er wie keine andere Gesamtplanung zuvor oder danach bereits eine detaillierte Ausgestaltung von Grund- und Aufrissen vor. Deshalb eignet sich der nach seinem Urheber benannte Schmiederplan in besonderer Weise für eine dreidimensionale Abbildung. Entwurf und Realisierung des Schmiederplans sind auch vor dem Hintergrund des aufsteigenden Nationalsozialismus zu sehen. Eine dreidimensionale Modellierung dieser Planung bietet somit losgelöst vom engeren Kontext dieser Arbeit auch die Möglichkeit zur kritischen Auseinandersetzung mit dem Thema Architektur bzw. Städtebau im Nationalsozialismus.

- Das Modell soll thematische 2D-Informationsebenen enthalten: Planwerke wie Flächennutzungsplan, Bebauungsplan, Stadtplan, Gesamtplanungen oder Luftbilder sollen integriert werden.

4.2 Informationssystem

Das Informationssystem soll bestimmte Bedingungen erfüllen. Diese werden in den folgenden Unterkapiteln spezifiziert. Sie lassen sich in inhaltliche, funktionale und technische Anforderungen gliedern. Die inhaltlichen Anforderungen leiten sich direkt aus dem Untersuchungsgebiet sowie aus Anknüpfungspunkten an die Planung ab, welche nachfolgend definiert werden. Die funktionalen und technischen Anforderungen werden vom Verfasser spezifiziert. Die Ansprüche an die Funktionalität basieren zu gleichen Teilen auf Gesprächen mit Verantwortlichen von Stadtplanungs- und Universitätsbauamt sowie verschiedenen Architekten, auf der Evaluierung anderer räumlicher Informationssysteme (BRAIL & KLOSTERMAN 2001; COORS & ZIPF 2005) und auf wissenschaftlich gesicherten Ansprüchen, die potentielle Nutzer solcher Systeme besitzen (BUCHHOLZ ET AL. 2005; FUHRMANN & MACEACHREN 2001).

4.2.1 Anknüpfungspunkte an die Planung

In Kapitel 2.3.2 wurde grob dargelegt, auf welchen Ebenen des Planungsprozesses das Informationssystem in erster Linie ansetzen soll. Dies sind die kommunale Ebene sowie die Ebene der Objektplanung. Im Folgenden wird erläutert, welche Teilprozesse auf diesen Planungsebenen mit welchen Mitteln unterstützt werden sollen.

Da das Informationssystem ein Instrument zur informellen Unterstützung von Planung und Entscheidungsfindung sein soll, wird die formelle Planung auf kommunaler Ebene abstrahiert und generalisiert betrachtet, um Anknüpfungspunkte des Informationssystems deutlicher herauszustellen. Der Planungsprozess, welcher das bauliche Erscheinungsbild des Universitätsgebiets „Im Neuenheimer Feld“ prägt, wird in Anlehnung an SELLE (2000, 61 ff.) als kommunikativer, sich erneuernder Prozess aufgefasst. Es wurde ein Konzept entworfen, welches den Kommunikationsprozess unabhängig von einzelnen, konkreten Planungsebenen in mehrere Stufen gliedert, die von verschiedenen Akteuren geprägt werden (Abb. 6). Das Informationssystem soll auf jeder dieser Stufen unterstützend wirken können.

Zu Beginn steht die Problemdefinition, bei der ein Akteur aus bestimmten Gründen Planungsbedarf anmeldet. Die dreidimensionale Visualisierung des Bestandes sowie die mit einzelnen Objekten verknüpften Informationen können dem Akteur als Argumentationshilfe bei der Schilderung des Problems dienen.

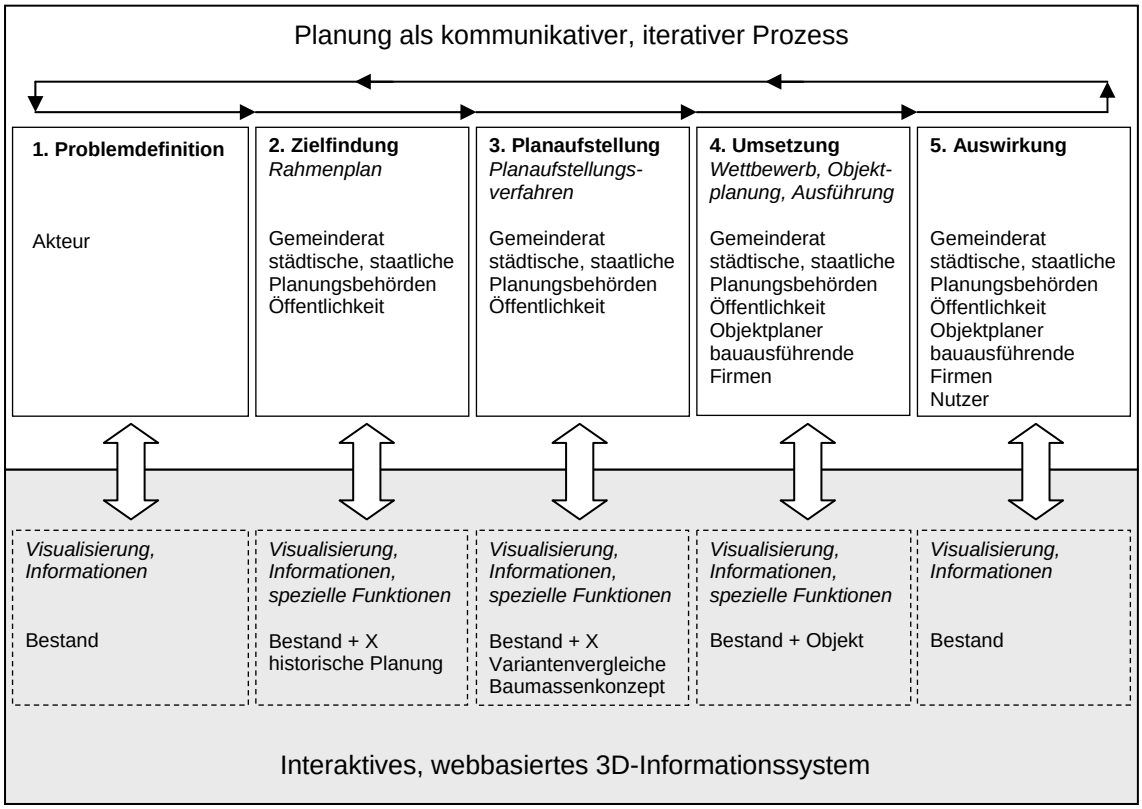

Abbildung 6: Anknüpfungspunkte des 3D-Informationssystems im Planungsprozess (eigene Darstellung)

Anschließend folgt die Zielfindung, bei der bereits mehrere Akteure, u.a. die Gemeinde bzw. Planungsbehörden sowie die Öffentlichkeit beteiligt sind. Die Zielfindung ist häufig ein informelles Verfahren, welches nicht offiziell im Baurecht geregelt ist. Zur Zielfindung werden beispielsweise Stadtteilrahmenpläne konzipiert. Diese nutzen Workshops als Instrument der Öffentlichkeitsbeteiligung (STADT HEIDELBERG 2002). Sofern hier baulich relevante Maßnahmenvorschläge entworfen werden, soll das Informationssystem mittels einfacher Visualisierung als interaktive Argumentations- und Anschauungshilfe dienen, die neben den unmittelbar an den Workshops beteiligten Bürgern auch der breiten Öffentlichkeit zur Verfügung steht. Im Rahmen der Ideenfin-

dung und Konsensbildung können im Informationssystem neben dem Bestand Vorhaben oder auch historische Planungen visualisiert werden und eigene, einfache Geometriemodelle aktiv eingebracht und variabel positioniert werden. Teile des Konzepts analoger Architekturmodelle sollen auf diese Weise digital zur Verfügung stehen (Abb. 7).

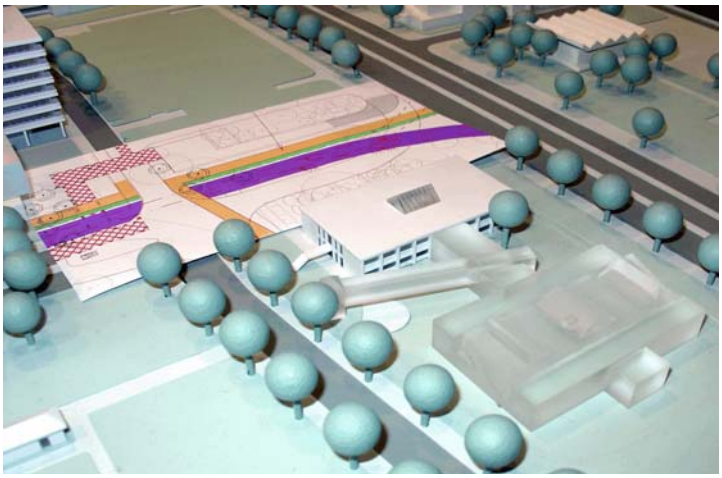

Abb. 7: Ausschnitt des im Universitätsbauamt stehenden analogen Architekturmodells; Bestand als weiße Kunststoffmodelle, Vorhaben als durchsichtige Acrylmodelle, Verkehrsplanung als in das Modell eingelegter, mehrfarbig bedruckter Kartonbogen (eigene Aufnahme)

Im darauf folgenden Abschnitt, der Planaufstellung, bei dem neben Gemeinde bzw. Planungsbehörden und der Öffentlichkeit weitere Akteure beteiligt sind, handelt es sich um ein rechtsförmliches Verfahren mit verbindlichen Vorgaben als Ergebnis (HANGARTER 1999, 24 ff.). Im formellen Planaufstellungsverfahren, worunter hier konkret die Bauleitplanung mit Flächennutzungsplan und Bebauungsplan verstanden wird, soll das Informationssystem sowohl in der Phase des Vorentwurfs wie auch des Entwurfs bei Entscheidungen helfen bzw. dazu dienen, die Diskussion zu versachlichen. Verschiedene Baumassenkonzepte für dasselbe Gebiet sollen als dreidimensionales Klötzchenmodell den Vergleich der Varianten veranschaulichen und die im Umgang mit Plänen ungeübte Öffentlichkeit aktiver in den Prozess einbeziehen. Neben der dreidimensionalen Visualisierung besteht auch hier die Möglichkeit, auf mit den Modellen verknüpfte Informationen zuzugreifen. Dies können beispielsweise erläuternde Textdokumente sein.

Nach Abschluss des Planaufstellungsverfahrens kann die Umsetzung erfolgen. Diese beinhaltet eine konkrete Objektplanung sowie deren bauliche Ausführung (SOMMER

1998, 20 ff.). Zu den bislang beteiligten Akteuren kommen in diesem Abschnitt schließlich Objektplaner und bauausführende Firmen hinzu. Für die Phase der Umsetzung soll das Informationssystem Möglichkeiten bereitstellen, geplante Einzelobjekte bereits vor Baubeginn oder Fertigstellung im späteren baulichen Umfeld zu visualisieren. Diese Funktionalität soll Architekten bei Wettbewerben und die Öffentlichkeit bei der qualifizierten Meinungsbildung unterstützen.
Als letzter Abschnitt des Kommunikationsprozesses ist die Auswirkung der Planung zu sehen. Hier sind alle in den vorherigen Abschnitten beteiligten Akteure versammelt. Mit der dreidimensionalen Visualisierung des Baubestands und dem Zugang zu mit einzelnen Objekten verknüpften Sachdaten und Informationen soll das Informationssystem Kommunikationsprozesse unterstützen, auf deren Grundlage der Planungsprozess von Neuem beginnt.

4.2.2 Inhaltliche und funktionale Anforderungen

Der Benutzer soll im Informationssystem auf bestimmte Geodaten zugreifen können. Dabei handelt es sich neben dreidimensionalen Geometriedaten auch um mit diesen verknüpfte Sachdaten. Diese Geodaten sollen es dem Nutzer ermöglichen, sich umfassend über die räumliche Lage und Funktion einzelner Gebäude im Untersuchungsgebiet zu informieren. Die Dateninhalte sollen es dem Benutzer darüber hinaus erlauben, einen Eindruck vom zukünftigen und historischen Planungsgeschehen im Untersuchungsgebiet zu bekommen. Vom Nutzer oder Dritten erstellte Geodaten sollen im Informationssystem angezeigt werden können. Die Geodaten sollen in einer Form präsentiert werden, die den Benutzer als virtuellen Betrachter vollständig in das Untersuchungsgebiet eintauchen lässt.

Das Informationssystem soll folgende Inhalte bereitstellen:

- Vollständiges 3D-Bestandsmodell: alle derzeit bestehenden Bauwerke des Untersuchungsgebiets.
- Generalisiertes 3D-Modell an das Untersuchungsgebiet angrenzender Gebiete: Bebauung von Wieblingen und Neuenheim.
- 3D-Modell des Geländes: Bett des Neckars und des Neckarkanals; charakteristische Abgrabungen im Bereich des Heidelberger Klinikrings.

- 3D-Modell der Infrastruktur: Straßen, Parkplätze und Gehwege; Grünflächen mit Baumkataster.
- 3D-Modelle ausgewählter Vorhaben und zukünftiger Planungen: Bauabschnitte des Heidelberger Klinikringes; Bauabschnitte des Physikalischen Instituts; Baumassenkonzept zur Gestaltung entlang der Berliner Straße; Varianten für eine fünfte Neckarquerung; Varianten für einen Nordzubringer über das Handschuhsheimer Feld; Varianten für eine Straßenbahntrasse durch das Neuenheimer Feld.
- 3D-Modelle ausgewählter historischer Planungen: die im Schmiederplan projektierten Kliniken, Verwaltungs- und Nebengebäude.
- Zweidimensionale Informationsebenen: Bestandsplan; Zielplanung; Flächennutzungsplan; Stadtplan; Luftbild; historische Pläne.
- Thematische Informationen zu einzelnen Objekten der genannten Modelle: allgemeine Informationen wie Name und Hausnummer von Gebäuden bzw. Einrichtungen; planungsrelevante Informationen wie Kosten, Fertigstellungstermine und Ziele von Vorhaben; Hyperreferenzen zu Webseiten mit weiteren Informationen sowie zu objektspezifischen Diskussionsforen im Internet zum Meinungsaustausch im Rahmen der Planungspartizipation.

Das Informationssystem soll folgende Funktionalitäten bieten:

- Freies Bewegen des Betrachters im 3D-Modell: hierzu sollen verschiedene Navigationsarten zur Verfügung gestellt werden. Der Nutzer kann auf diese Weise alle auch in der Realität möglichen Positionen und Blickwinkel im 3D-Modell einnehmen. Darüber hinaus soll es ihm möglich sein, Blickwinkel einzunehmen, die in der Realität schwierig oder gar nicht zu realisieren sind.
- Gezielter Aufruf bestimmter Positionen im 3D-Modell: der Nutzer soll bestimmte vorgegebene Positionen in der Szene einnehmen können. Diese Positionen gleichen virtuellen Kameras, die bestimmte Einzelgebäude des 3D-Modells oder planungstechnisch interessante Bereiche zeigen.
- Absolute Positionierung des Avatars über eine 2D-Karte: es soll möglich sein, über einen zweidimensionalen Lageplan des Untersuchungsgebiets die Position des virtuellen Betrachters zu steuern.

- Freies Wählen des Sichtfeldes: der Nutzer soll entscheiden können, ob die Darstellung der dreidimensionalen Szene dem Sichtverhalten eines Menschen entsprechen soll, oder ob er eine Weitwinkel- oder Teleansicht der 3D-Modelle wünscht, wie sie auch fotographisch erzeugt werden können. Das Sichtfeld soll zur Laufzeit des Systems verändert werden können. Der Nutzer soll permanent Rückmeldung über das von ihm eingestellte Sichtfeld in Form eines entsprechenden Parameters erhalten.
- Wählen der Darstellungsart: das System soll dem Nutzer verschiedene Möglichkeiten der Darstellung der 3D-Modelle erlauben. Die Modelle sollen neben der Darstellung mit Fotos der Gebäudefassaden auch untexturiert dargestellt werden können. Ebenso soll eine Drahtgitterdarstellung zur Verfügung stehen.
- Zusätzliche Lichtquelle: um Innenräume von Gebäuden aufzuhellen, soll eine zusätzliche an den virtuellen Betrachter gebundene Lichtquelle zur Verfügung stehen.
- Gezielte Abfrage von thematischen Informationen: durch Anklicken einzelner 3D-Modelle soll der Nutzer thematische Informationen abfragen können. Es soll die Möglichkeit gegeben werden, für die Abfrage bestimmte Kategorien vorzugeben, um gezielt spezifische Informationen zu einem Objekt zu erhalten.
- Ein- und Ausblenden von 3D-Modellen und 2D-Informationsebenen: der Nutzer soll die Modelle ausgewählter Vorhaben und Planungen in das Bestandsmodell einblenden können. Ebenso sollen die 2D-Informationsebenen in der Darstellung ein- oder ausgeschaltet werden können. Auf diese Art erhält der Nutzer die Möglichkeit verschiedene Inhalte zu kombinieren.
- Dynamische Integration fremder Geometriedaten: das System soll die Möglichkeit bieten, vom Nutzer oder Dritten erstellte 3D-Modelle oder 2D-Informationsebenen zur Darstellung in das System zu integrieren. Dies soll zur Laufzeit des Systems ohne Notwendigkeit eines Neustarts möglich sein. Hierzu soll das Konzept analoger Architekturmodelle digital abgebildet werden.
- Dynamisch-variable Integration vorgegebener Geometriedaten: der Nutzer soll Zugriff auf eine Auswahl von Typenbauten (vgl. Kap. 3.2.2) erhalten, die für das Untersuchungsgebiet charakteristisch sind. Diese soll er dynamisch und zur Laufzeit des Systems der aktiven Szenendarstellung hinzufügen können. Die

räumliche Position dieser Typenbauten soll dabei variabel bleiben, d.h. der Nutzer kann diese Gebäude beliebig im 3D-Modell verschieben.

- Abspeichern der aktuellen Ansicht und Inhalte: der Nutzer soll die Möglichkeit haben, die aktuelle Darstellung des 3D-Modells als Bilddatei abzuspeichern, um sie beispielsweise für Präsentationszwecke zu verwenden. Ebenso soll es möglich sein, Inhalte der 3D-Szene abzuspeichern, um sie zu einem späteren Zeitpunkt wieder aufrufen zu können. Dies ist insbesondere im Zusammenhang mit dynamisch zusätzlich in die Szene eingebrachten Objekten interessant.
- Rückmeldung über die Position des virtuellen Betrachters: um dem Nutzer die Navigation in der virtuellen Szene zu erleichtern, soll die aktuelle Position auf einer 2D-Karte angezeigt werden. Zudem soll die Ausgabe der Position in Gauß-Krüger-Koordinaten sowie Höhe über Grund zur Verfügung stehen. Der Nutzer soll darüber hinaus über die Blickrichtung informiert werden. Dies soll grafisch über einen Pfeil auf der 2D-Karte sowie als Zahlenwert erfolgen.

4.2.3 Technische Anforderungen

Die technischen Anforderungen an das Systems leiten sich zum Teil direkt aus dem Anspruch einer möglichst ubiquitären Verfügbarkeit ab. Um diesem Anspruch zu genügen, wurde das Internet als Plattform für das Informationssystem ausgewählt.
Eine der wichtigsten Anforderungen ist die Basierung des Systems auf international standardisierten Technologien. Im Gegensatz zu proprietären, an bestimmte kommerzielle Hersteller gebundenen Technologien garantieren Standards den freien Zugriff auf die Spezifikation einer Technologie. Die Pflege und Weiterentwicklung der Technologie ist unabhängig von den Interessen einzelner Firmen oder Hersteller. Standards versprechen somit langfristig eine hohe Investitionssicherheit zumal bei ihrer Nutzung üblicherweise keine Lizenzgebühren an bestimmte Hersteller fällig werden.

Das Informationssystem soll folgende technische Anforderungen erfüllen:

- Ubiquitäre Verfügbarkeit: das System soll das Internet als Plattform nutzen und dadurch nahezu orts- und zeitunabhängig zur Verfügung stehen.
- Basierung auf Standards: das System soll so weit wie möglich von international anerkannten Standardisierungsorganisationen spezifizierte Technologien nutzen.

- Echtzeitfähigkeit: Manipulationen und Interaktionen des Nutzers in bzw. mit dem 3D-Modell sollen möglichst ohne spürbare Verzögerung ausgeführt werden. Dies soll durch eine spezielle Systemarchitektur erreicht werden, bei der der Rechner des Nutzers einen Großteil der Datenverarbeitung übernimmt.
- Modularer Aufbau: einzelne Funktionalitäten des Informationssystems sollen unabhängig voneinander sein, so dass eine beliebige Kombination oder ein Weglassen einzelner Funktionalitäten nicht die Funktionsfähigkeit des Gesamtsystems beeinflusst. Hierdurch wird eine vergleichsweise einfache und schnelle Entwicklung zusätzlicher Funktionen in Form weiterer Module ermöglicht.
- Intuitive Bedienbarkeit: die Nutzung des Systems soll auch für Nicht-Fachleute möglich sein. Deshalb soll das Informationssystem in einen Webbrowser integriert werden, dessen Bedienungskonzept einer breiten Öffentlichkeit bekannt ist. Der Zugriff auf einzelne Funktionen des Systems soll sich zudem an Metaphern orientieren, die dem potentiellen Nutzer vom Umgang mit anderen Software-Applikationen vertraut sind.
- Indizierung einzelner 3D-Modelle mit eindeutigen IDs: jedes 3D-Modell soll eine eigene ID besitzen, um die gezielte Zuordnung von Sachdaten zu ermöglichen. Die IDs sollen an zentraler Stelle in einem einzigen Dokument verwaltet werden können.
- Möglichkeit zur externen Referenzierung von 3D-Daten: das Informationssystem soll die Möglichkeit bieten, 3D-Modelle auch dezentral zu verwalten. So sollen Aufwand für Pflege und Wartung der 3D-Modelle minimiert werden. Gleichzeitig wird hierdurch das Einbinden der Daten von Dritten erleichtert. Die IDs für diese Modelle sollen trotzdem an zentraler Stelle verwaltbar bleiben.
- Möglichkeit zum Offline-Betrieb: gewisse Grundfunktionalitäten des Informationssystem sollen auch als eigenständige Applikation unabhängig vom Internet nutzbar sein. Dadurch soll ein Einsatz auf lokalen, nicht an das Internet angebundenen Rechnern ermöglicht werden. Dies soll durch die bereits im Punkt „Echtzeitfähigkeit“ erwähnte Systemarchitektur gewährleistet werden.
- Skalierbarkeit: das System soll sowohl hinsichtlich seiner Inhalte als auch der Anzahl der Nutzer skalierbar sein. Das bedeutet, dass sich die Erweiterung des Inhalts oder die Erhöhung der Anzahl gleichzeitiger Nutzerzugriffe nicht

nachteilig auf die Performanz des Systems auswirken sollen. Dies soll bei der Gestaltung der Inhalte sowie der Systemarchitektur berücksichtigt werden.

Eine Verwaltung von 3D-Daten in einer Datenbank wie sie in anderen Projekten umgesetzt wurde (SCHILLING ET AL. 2005, 250 ff.), wird in dieser Arbeit nicht angestrebt, da das zu verwaltende Datenvolumen bzw. die Anzahl der Einzelmodelle vergleichsweise gering ist.

5 Digitale Abbildung des Untersuchungsgebiets

Die dreidimensionale Modellierung des Heidelberger Universitätscampus ist eines der beiden Kernziele dieser Arbeit. Die nachfolgenden Unterkapitel erläutern, welche Quellen dafür herangezogen wurden und welche methodische Vorgehensweise für die Modellierung gewählt wurde, um die in Kapitel 4.1 aufgestellten Anforderungen zu erfüllen.

5.1 Quellen

Wie in Kapitel 4.1 postuliert und begründet wurden für die Modellierung bestehende Datenquellen herangezogen. Diese standen in unterschiedlicher Form zur Verfügung. Zusätzlich wurden umfangreiche eigene Erhebungen durchgeführt.

5.1.1 Digitale Daten

Als Basis für die Grundrisse der bestehenden Gebäude und deren relative Lage zueinander diente der aktuelle Bestandsplan des Heidelberger Universitätsbauamts (UNIVERSITÄTSBAUAMT HEIDELBERG 2005). Dieser als Vektor-Datei vorliegende CAD-Plan enthält rund 30 verschiedene Informationsebenen. Neben den aus den Dachaufsichten abgeleiteten Gebäudegrundrissen wurden Elemente der Infrastruktur wie Straßen, Parkplätze oder Fußwege sowie Grün- und Wasserflächen und ein Baumkataster als einzelne Informationslayer aus diesem Dokument genutzt. Abbildung 8 zeigt einen Ausschnitt dieses CAD-Plans.
Für die zukünftig geplanten Gebäude und Infrastrukturelemente dienten zum einen die aktuelle Zielplanung des Universitätsbauamts (UNIVERSITÄTSBAUAMT HEIDELBERG 2004b) sowie zum anderen hoch detaillierte Pläne von Architekturbüros als Quellen für die dreidimensionale Modellierung (UNIVERSITÄTSBAUAMT HEIDELBERG 2003-2005). Ausschnitte dieser Vektordaten sind in Abbildung 8 zu sehen.

Die Gebäudehöhen basieren zum größten Teil auf einem digitalen Rasterplan älteren Datums (UNIVERSITÄTSBAUAMT HEIDELBERG 1993). Dieser enthält neben dem damaligen Baubestand absolute Höhenwerte über Normalnull für einzelne Dachflächen der Gebäude sowie Höhenangaben von markanten Geländepunkten. Abbildung 8 zeigt einen Ausschnitt aus diesem Plan.
Als weitere digitale Quellen wurden der Stadtplan und Luftbilder von Heidelberg (VERMESSUNGSAMT HEIDELBERG 2002 & 2003) sowie der aktuelle Flächennutzungsplan 2015/2020 (NACHBARSCHAFTSVERBAND HEIDELBERG-MANNHEIM 2006) herangezogen.

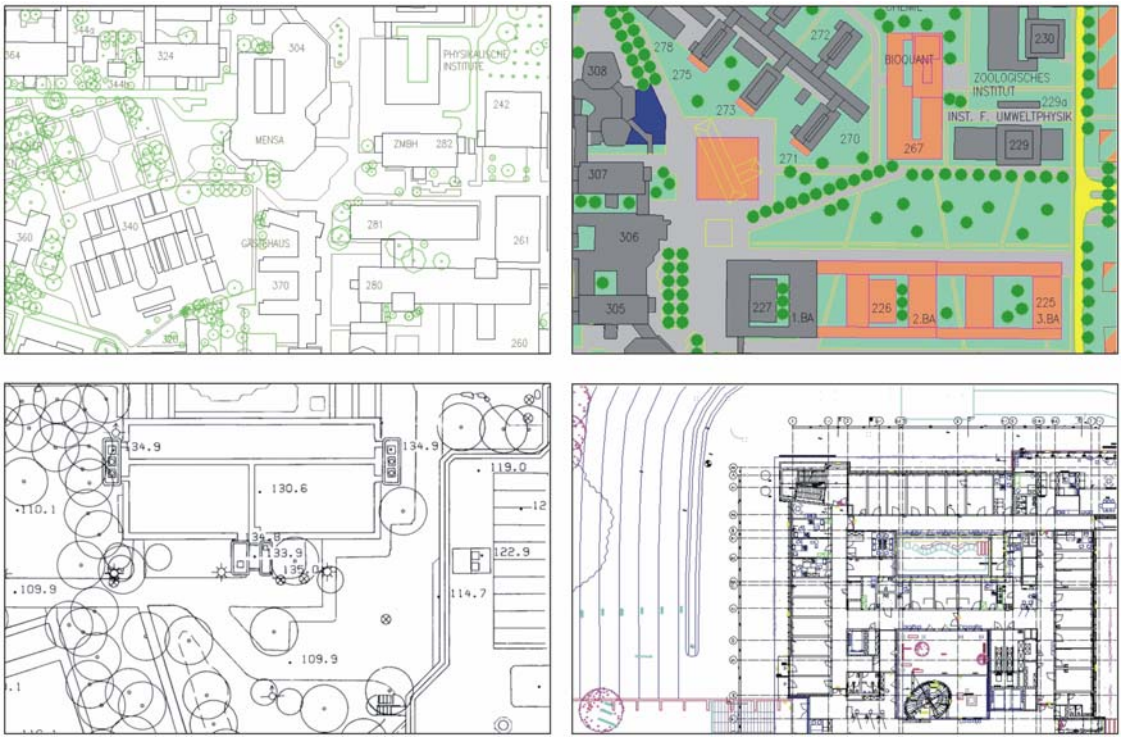

Abb. 8: Verschiedene digitale Planwerke des Untersuchungsgebiets: Ausschnitte von Bestands- und Zielplanung (oben links, oben rechts); Ausschnitte des Rasterplans mit absoluten Höhen und eines detaillierten Architekturplans (unten links, unten rechts). (eigene Darstellung nach UNIVERSITÄTSBAUAMT HEIDELBERG 2003-2005)

5.1.2 Analoge Daten

Insbesondere für die Abbildung der historischen Planung standen ausschließlich analoge Datenquellen zur Verfügung. Zur Modellierung des Schmiederplans wurden ein

kleinmaßstäblicher Lageplan (vgl. Abb. 2) sowie zahlreiche Abbildungen eines Architekturmodells (Abb. 9) der projektierten Gebäude analysiert. Zusätzlich wurde auf sich im Archiv des Universitätsbauamts befindende Aufrisszeichnungen aus dem Jahr 1933 im Maßstab 1:100 zurückgegriffen (vgl. auch MÜLLER 1998, 176 ff.).

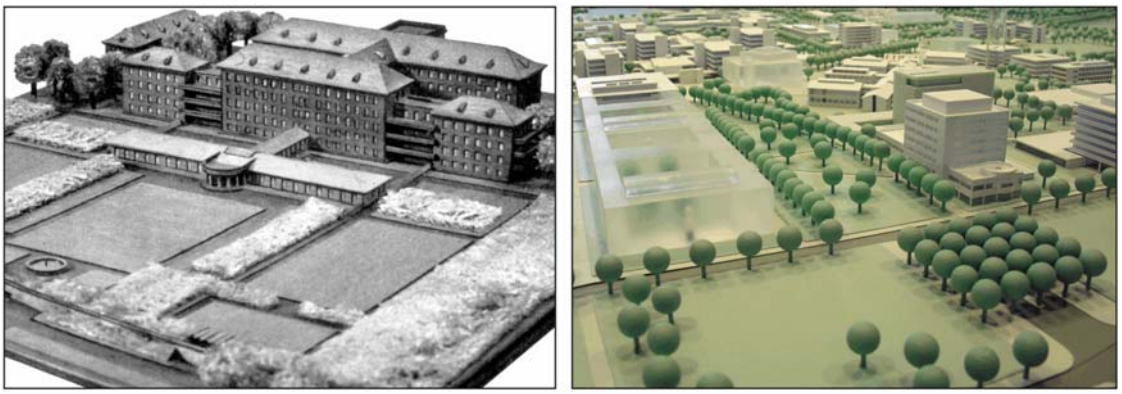

Abb. 9: links: Ausschnitt aus dem Architekturmodell des Generalbebauungsplans von 1932 (SCHMIEDER 1936, 22); rechts: Ausschnitt aus dem Architekturmodell des Universitätsbauamts (eigene Aufnahme)

Für die Abbildung der zukünftigen Vorhaben wurden für bestimmte Gebäude analoge Architekturmodelle (Abb. 9) zur Auswertung herangezogen, da die entsprechenden digitalen Pläne derzeit noch zu wenig Informationen beispielsweise über die Gebäudehöhen enthalten. Die Modellierung geplanter Varianten der zukünftigen Verkehrsinfrastruktur basiert auf analogen Planwerken des Heidelberger Universitätsbauamts (UNIVERSITÄTSBAUAMT HEIDELBERG 2002) sowie auf umfangreichen Textquellen, Karten und Plänen einer von der Stadt Heidelberg in Auftrag gegebenen Untersuchung (STADT HEIDELBERG 2005b). Zusätzlich wurden Gespräche mit Planungsverantwortlichen der Stadt und des Universitätsbauamts geführt.

5.1.3 Eigene Erhebungen

Die oben genannten digitalen und analogen Quellen dienten in erster Linie dazu, konkrete Informationen über die geometrische Beschaffenheit der zu modellierenden Objekte zu erhalten. Die Erscheinung der Objekte, das heißt die Ausstattung mit aussagekräftigen, realistischen Texturen basiert auf eigenen Erhebungen. Hierzu wurden rund 1200 fotographische Aufnahmen einzelner Gebäude und Gebäudekomplexe gemacht. Diese wurden ausgewertet und aufwendig bearbeitet (vgl. Kap. 5.3.4).

Um charakteristische Abgrabungen im Bereich der Medizinischen Klinik in der Modellierung berücksichtigen zu können, wurden entsprechende Daten im Gelände erhoben. Hierzu wurde ebenso wie für die Georeferenzierung einzelner Gebäude ein Differential-GPS eingesetzt. Dieses integriert terrestrische Korrektursignale, was die Positionsgenauigkeit auf ca. einen Meter sichert. Die Genauigkeit wurde anhand der Position von Vermessungspunkten verifiziert.
Um zweidimensionale Daten als Grundlage für die weitere Verarbeitung zu gewinnen, wurden Karten bzw. Luftbilder digitalisiert. Die an das Untersuchungsgebiet angrenzende Bebauung von Wieblingen und Neuenheim wurde auf Grundlage oben genannter Luftbilder manuell digitalisiert. Aus der Topographischen Karte 1:25.000 für Baden-Württemberg (LANDESVERMESSUNGSAMT BADEN-WÜRTTEMBERG 2002) wurden 2.5D-Daten für das Neckarbett und den Neckarkanal digitalisiert.
Gebäudehöhen, die nicht aus oben genannten Planwerken entnommen werden konnten, wurden vor Ort durch Messungen der Höhe eines Stockwerks und anschließender Extrapolation über die Gesamtzahl der Stockwerke des betreffenden Gebäudes ermittelt.

5.2 Aufbereitung und Verwaltung der Basisdaten

In Kapitel 4.1.1 wurde postuliert, dass die Prozesskette für die Aufbereitung und Verwaltung der dem 3D-Modell zugrunde liegenden Daten so konzipiert sein soll, dass die Daten möglichst anwendungsneutral zur Verfügung stehen. Dadurch soll erreicht werden, dass die Geodaten auch anderen Anwendungszwecken als dem in dieser Arbeit im Fokus stehenden zugeführt werden können. Um diese Anforderung zu erfüllen, wurden sämtliche zweidimensionalen Geometriedaten, unabhängig ob es sich um Raster- oder Vektordaten handelte, der Verwaltung in einem Geographischen Informationssystem zugeführt. Hierfür wurde die Software ArcGIS des Marktführers ESRI ausgewählt.
Die digital zur Verfügung stehenden CAD-Pläne wurden zuvor mit der Software AutoCAD, der am häufigsten eingesetzten Entwurfssoftware für Architekten und Ingenieure, in einzelne Informationsebenen separiert. Auf diese Weise konnten ausgewählte Informationslayer über entsprechende Austauschdatenformate als Vektordaten in ArcGIS importiert werden. Weitere Planwerke und Luftbilder wurden zunächst in Bildbearbeitungsprogrammen aufbereitet und anschließend als Rasterdaten in ArcGIS importiert. Mit diesen beiden Methoden wurden insgesamt etwa 40 verschiedene Layer in das GIS portiert. In einem weiteren Schritt wurden diese Layer georeferenziert. Dies geschah unter Nutzung der dafür in ArcGIS zur Verfügung stehenden Werkzeuge und

unter Zuhilfenahme eines am Geographischen Institut verfügbaren und bereits referenzierten Datensatzes, welcher räumliche Überschneidungen und somit Vergleichsobjekte enthielt. Für einige Gebäude wurden darüber hinaus eigene, mit einem Differential-GPS erhobene Referenzdaten verwendet. Diese wurden ebenfalls zur Absicherung des Vergleichdatensatzes herangezogen. Als Bezugssystem für die Georeferenzierung wurde das Gauß-Krüger-Koordinatensystem ausgewählt, da dies üblicherweise für amtliche Karten- und Planwerke Verwendung findet.

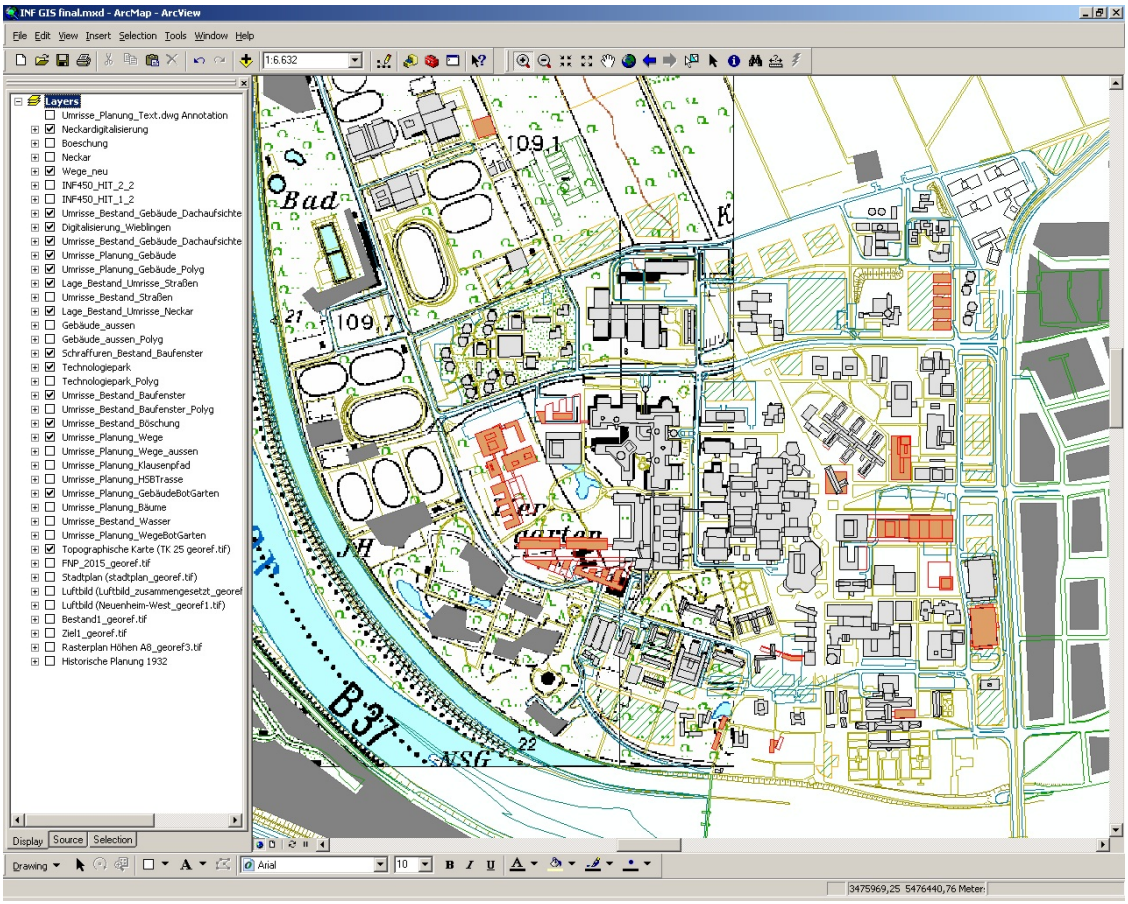

Abb. 10: GIS für das Untersuchungsgebiet mit verschiedenen Informationslayern (Screenshot aus ArcGIS; eigene Darstellung)

Abbildung 10 zeigt ausgewählte Informationsebenen des in dieser Arbeit für das Untersuchungsgebiet aufgebauten Geographischen Informationssystems. Die Integrierung und Verwaltung der beschriebenen Inhalte mittels GIS erlaubt eine breite Verwendung der enthaltenen Daten. Die verschiedenen Layer sind untereinander beliebig kombi-

nierbar. Eine Verknüpfung der Geometriedaten mit Sachdaten ist möglich und erlaubt vielfältige Analysen für verschiedenste Fragestellungen. ArcGIS gestattet den Export von Daten in verschiedenen Formaten. Dieser Weg wurde eingeschlagen, um die Daten für die in dieser Arbeit im Mittelpunkt stehende dreidimensionale Modellierung des Untersuchungsgebiets zu nutzen. Eine weitere Anwendungsmöglichkeit ist die Nutzung der Geodaten in einem Open Source WebGIS, wie es vom Verfasser beispielsweise in ACHSTETTER ET AL. (2006) realisiert wurde. Abbildung 11 zeigt in schematischer und vereinfachter Form die Prozesskette, welche zur Erhebung, Verarbeitung und Verwaltung der verschiedenen Daten konzipiert wurde.

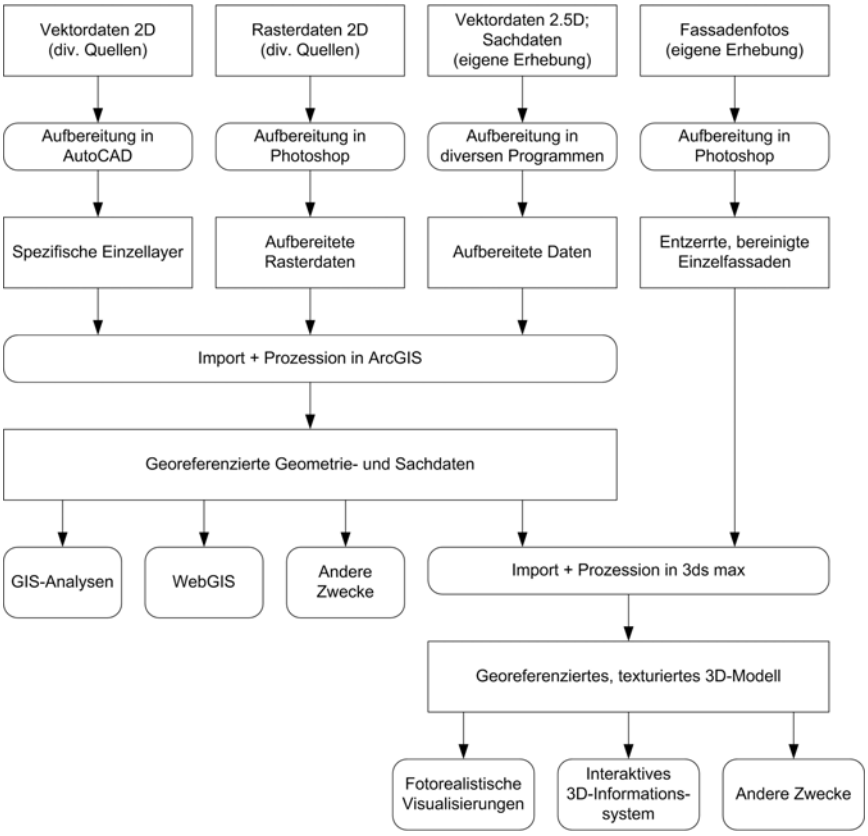

Abbildung 11: Prozesskette mit Ausgangsdaten, verschiedenen Schritten der Datenprozession, Zwischen- und Endergebnissen sowie möglichen Anwendungszwecken (eigene Darstellung)

Die wichtigsten Layer, die das in dieser Arbeit aufgebaute Geographische Informationssystem zur Verfügung stellt, sind nachfolgend unter Angabe des Datentyps und der Quelle aufgelistet:

- Bestand Umrisse Gebäude (Vektordaten; UNIVERSITÄTSBAUAMT HEIDELBERG 2005)
- Bestand Umrisse Wege (dto.)
- Bestand Umrisse Straßen (dto.)
- Bestand Umrisse Parkplätze (dto.)
- Bestand Umrisse Grünflächen (dto.)
- Bestand Umrisse Wasserflächen (dto.)
- Bestand Baumkataster (dto.)
- Bestand Neckarbett und Kanal (Vektordaten; eigene Digitalisierung basierend auf LANDESVERMESSUNGSAMT BADEN-WÜRTTEMBERG 2002)
- Bestand Abgrabung Medizinische Klinik (Vektordaten; eigene digitale Erhebung im Gelände)
- Bestand Randbebauung Neuenheim und Wieblingen (Vektordaten; eigene Digitalisierung basierend auf VERMESSUNGSAMT HEIDELBERG 2002)
- Plan mit absoluten Höhenangaben (Rasterdaten; UNIVERSITÄTSBAUAMT HEIDELBERG 1993)
- Luftbild Neuenheimer Feld (Rasterdaten; VERMESSUNGSAMT HEIDELBERG 2003)
- Stadtplan Heidelberg (Rasterdaten; VERMESSUNGSAMT HEIDELBERG 2002)
- Flächennutzungsplan 2015/2020 (Rasterdaten; NACHBARSCHAFTSVERBAND HEIDELBERG-MANNHEIM 2006)
- TK 25 Bereich Neuenheimer Feld und angrenzende Gebiete (Rasterdaten; LANDESVERMESSUNGSAMT BADEN-WÜRTTEMBERG 2002)
- Historischer Generalbebauungsplan von 1932 (Rasterdaten; SCHMIEDER 1936)
- Planung Umrisse Gebäude (Vektordaten; UNIVERSITÄTSBAUAMT HEIDELBERG 2004)
- Planung Umrisse Wege (dto.)
- Planung Umrisse Straßen (dto.)

- Planung Umrisse Parkplätze (dto.)
- Planung Umrisse Grünflächen (dto.)
- Planung Umrisse Wasserflächen (dto.)
- Planung Baumkataster (dto.)

5.3 Dreidimensionale Modellierung

Dreidimensionale Geoobjekte können ebenso wie zweidimensionale Geodaten spezifische Parameter besitzen. Nachfolgend wird erläutert, welche charakteristischen Komponenten in der 3D-Modellierung Berücksichtigung finden, und welches konkrete Konzept entworfen wurde, um die in Kapitel 4.1.1 aufgestellten Anforderungen zu erfüllen.

5.3.1 Typisierung der Komponenten

Zweidimensionale Geoobjekte in Geographischen Informationssystemen zeichnen sich nach BILL (1999a, 10 ff.) durch verschiedene Charakteristika aus. Die Geometrie beschreibt die räumliche Lage und Ausdehnung des Objekts in einem gegebenen Bezugsraum. Die Topologie eines Objekts definiert dessen räumliche Beziehung zu anderen Objekten. Die Nachbarschaft zu Objekten, das Enthalten weiterer Objekte oder die Überschneidung mit anderen Objekten können beispielsweise Merkmale topologischer Beziehungen sein. Die Semantik eines Objekts beschreibt dessen thematische Ausprägungen, also die Verknüpfung mit Sachdaten oder Attributen. Der Schlüssel eines Objekts dient als Identifikator der eindeutigen Zuweisung von geometrischen und thematischen Inhalten.
Auch für dreidimensionale Geodaten wie beispielsweise 3D-Stadtmodelle besteht grundsätzlich der Bedarf, die genannten charakteristischen Aspekte abbilden zu können, um eine multifunktionale Verwendbarkeit der Daten zu gewährleisten. Derzeit bestehende Datenformate bieten jedoch insbesondere hinsichtlich der Repräsentation von Topologie und Semantik nur unzureichende Möglichkeiten (GRÖGER & KOLBE 2005, 56). Auch einheitliche oder gar standardisierte Verfahren zur Erhebung, Verwaltung und Pflege von 3D-Stadtmodellen existieren bislang nicht. Entwurf, Spezifizierung und Evaluierung geeigneter Datenmodelle sind aus diesem Grund Gegenstand aktueller wissenschaftlicher Forschung (OGC 2006; ZIPF 2007). Eine laufende Unter-

suchung am Geographischen Institut der Universität Heidelberg analysiert Verfahren, die derzeit bei der Erstellung von 3D-Stadtmodellen in großen Kommunen zum Einsatz kommen. Erste Zwischenergebnisse bestätigen, dass gegenwärtig eine Vielzahl unterschiedlicher Herangehensweisen existiert (GÖBEL 2007).
Für eine wirklichkeitsnahe Repräsentation eines dreidimensionalen Geoobjekts wird neben den oben angesprochenen, zum Teil optionalen Aspekten eine weitere Komponente zwingend benötigt. Die graphische Erscheinung bzw. Darstellung des Objekts muss definiert werden. Hierzu kann ein 3D-Objekt mit einer Textur ausgestattet werden (vgl. Kap. 2.3.4).
Als grundsätzlich mögliche Komponenten dreidimensionaler Geodaten ergeben sich somit zusammenfassend die Geometrie, die graphische Erscheinung, die Topologie, die Semantik und der Identifikator. Topologie und Semantik können wie erwähnt mit derzeit verfügbaren Konzepten nicht oder nur unzureichend abgebildet werden. Die Komponenten Geometrie und graphische Erscheinung sind hingegen zwingend notwendig und können mit den gängigen Entwicklungsumgebungen zur Erstellung dreidimensionaler Modelle realisiert werden. Solche Entwicklungswerkzeuge erlauben in der Regel die Vergabe von eindeutigen Namen für einzelne geometrische Bestandteile eines Modells. Diese können als Identifikator dienen und über Referenzlisten die Zuordnung von bzw. Verknüpfung mit Sachdaten ermöglichen. Diese Möglichkeit wurde in vorliegender Arbeit genutzt, um insbesondere für Verwendung des 3D-Modells im 3D-Informationssystem eine semantische Komponente zu realisieren (vgl. Kap. 6.5.7).
Als Softwareumgebung für die Erstellung des 3D-Modells wurde das Programm 3ds max der Firma Autodesk ausgewählt. Dies ist die im professionellen Umfeld weltweit am häufigsten eingesetzte Software im Bereich 3D-Modellierung und Animation. Das in dieser Software genutzte Dateiformat 3DS gilt als einer der Quasi-Standards zum Austausch von 3D-Daten (FREIWALD & JANY 2005, 154).

5.3.2 Komponentengewichtung und Modellierungskonzept

Während einige der im vorangegangenen Kapitel beschriebenen Komponenten entweder Teil eines Modells sind oder nicht, können die Bestandteile Geometrie und graphische Erscheinung bzw. Texturierung unterschiedlich starke Berücksichtigung im 3D-Modell finden. Hierauf basiert das in dieser Arbeit verwendete Modellierungskonzept.
In Kapitel 4.1.1 wurde als Prämisse für das 3D-Modell eine möglichst universelle Nutzungsmöglichkeit postuliert. Als Gegenpole wurden einerseits die Verwendung für

fotorealistische Visualisierung, nachfolgend Visualisierungsmodell genannt, und andererseits die Anwendung in interaktiven Online-Applikationen, nachfolgend Webmodell genannt, beschrieben. Ziel war es, das dreidimensionale Campusmodell so zu konzipieren, dass es unterschiedlichen Anwendungsmöglichkeiten zugeführt werden kann, ohne dass sich hierdurch notwendige Kompromisse negativ auf die spezifische Anwendung auswirken. Während die interaktive Verwendung als Webmodell neben gewissen anderen technischen Anforderungen ein möglichst geringes Datenvolumen notwendig macht, muss die Verwendung als Visualisierungsmodell in erster Linie inhaltliche Anforderungen, nämlich hohen Detailreichtum des Modells, erfüllen, welche den technischen Anforderungen des Webmodells entgegengesetzt sind. Um beide Anforderungen erfüllen zu können, wurde ein Modellierungskonzept entworfen, welches sowohl dem Webmodell als auch dem Visualisierungsmodell gerecht wird.

Hinsichtlich ihrer optischen Erscheinung setzen sich computergenerierte 3D-Modelle aus zwei Komponenten zusammen. Dies sind die Geometrie, die die äußere Form des Modells bestimmt, und die Textur, welche als bildhaftes Element das Aussehen der geometrischen Oberfläche bestimmt. Es wurde ein Konzept entworfen, welches den für das Visualisierungsmodell notwendigen hohen Detailreichtum nicht über eine im Vergleich zum Webmodell aufwendigere Geometrie erreicht, sondern über höher auflösende Texturen. Umgekehrt wird die maßgebliche Anforderung des Internetmodells an ein geringes Datenvolumen dadurch erreicht, dass die sehr hoch auflösenden Texturen des Visualisierungsmodells automatisiert heruntergerechnet werden, um dadurch weniger Speicherplatz in Anspruch zu nehmen. Der Weg, die Texturkomponente je nach Anwendungszweck des 3D-Modells automatisiert zu verändern, hat den entscheidenden Vorteil, dieselbe Geometriekomponente für beide Modelle anwenden zu können. Zwar existieren auch für die Vereinfachung der Geometrie automatische Verfahren, diese sind jedoch zum einen extrem rechenaufwendig und zum anderen nach eigener Erfahrung für den Einsatz bei hoch detaillierten 3D-Stadtmodellen noch nicht gänzlich ausgereift. Die Methoden zur Vereinfachung der Geometrie sollen die Anzahl der Polygone eines 3D-Objekts reduzieren, ohne den visuellen Eindruck des Objekts beim Betrachter spürbar zu verändern. Polygone bilden als einzelne Flächen die geometrischen Grundelemente eines 3D-Objekts. Solche Polygonsimplifizierungsverfahren sind ebenfalls Gegenstand aktueller Forschung (ZACH ET AL. 2005, 206). Eine besondere Herausforderung besteht dabei darin, die menschliche visuelle Wahrnehmung von dreidimensionalen Objekten zu verstehen (ALEXA 2006, 3).

Das in dieser Arbeit genutzte Konzept, welches die Texturkomponente stärker gewichtet, berücksichtigt auch die spezielle Bauweise der Gebäude des Untersuchungsgebiets. Da die geometrische Form vieler Bauten aufgrund der standardisierten Typenbauweise (vgl. Kap. 3.2.2) eine mathematisch wie modelliertechnisch relativ einfach zu beschreibende ist und somit per se dazu beiträgt, das Datenvolumen gering zu halten, kommt der Texturkomponente eine höhere Bedeutung im Modellierungsprozess zu. Die standardisierte Bauweise ermöglicht zudem die Instanzierung, vereinfacht ausgedrückt die Mehrfachnutzung, gewisser Gebäudetypen. Dieser Umstand kommt den Ansprüchen des Internetmodells zu Gute, da die Instanzierung es erlaubt, bestimmte Gebäude nur einmal über das Internet zum Nutzer zu übertragen, aber mehrfach als geometrische Repräsentation im Gesamtmodell zu verwenden.

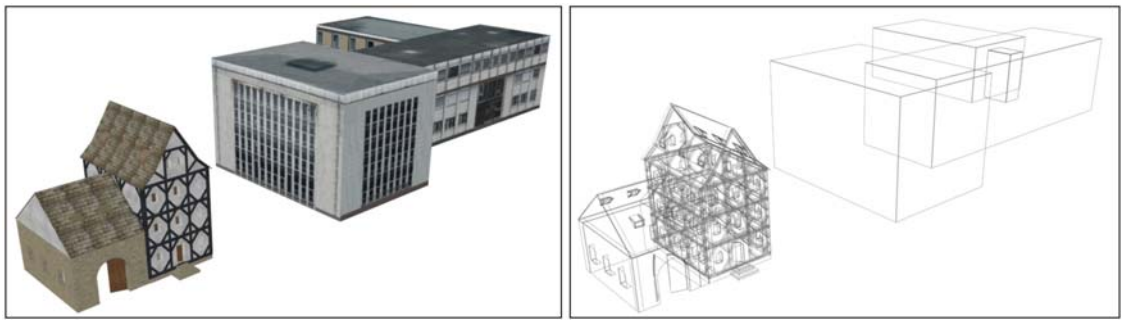

Abb. 12: Gegenüberstellung verschiedener Modellierkonzepte für 3D-Modelle (eigene Modelle und Darstellung)

In Abbildung 12 werden zwei 3D-Modelle, die auf unterschiedlichen Konzepten basieren gegenübergestellt, um die Besonderheiten der verschiedenen Ansätze zu verdeutlichen. Bei den Modellen handelt es sich zum einen um ein Fachwerkensemble und zum anderen um das Geographische Institut der Universität Heidelberg. Im linken Teil der Abbildung werden die beiden Modelle in ihrer texturierten Ansicht gezeigt, im rechten Teil in einer Drahtgitterdarstellung, die die den Modellen zugrunde liegende Geometrie sichtbar macht. Für das Fachwerkhaus wurde ein Großteil konstruktiver Elemente wie Fenster, Dachgauben oder das Fachwerk vollständig durch die Geometrie abgebildet. Die Polygonanzahl ist dadurch mit über 5000 Dreieckspolygonen sehr hoch. Zusätzlich wurden die unterschiedlichen Bauteile noch mit rund 20 verschiedenen Texturen versehen. Der Speicherplatz und Datenverwaltungsaufwand für großflächige Modelle dieser Art ist immens hoch. Für interaktive Echtzeitdarstellungen sind auf einem solchen geometrielastigen Konzept basierende Modelle grundsätzlich nicht geeignet.

Für das Institutsgebäude wurde der in dieser Arbeit konzipierte Ansatz gewählt, der eine stärkere Gewichtung der Texturen in den Vordergrund stellt. Konstruktive Fassadenelemente werden ausschließlich durch die Textur abgebildet. Die Geometrie kann dadurch sehr schlank gehalten werden. Zur Darstellung der Geometrie des Modells sind gerade einmal vier Quader bzw. 48 Dreieckspolygone notwendig. Zusätzlich kommen 14 hochauflösende Texturen zum Einsatz, die für verschiedene Anwendungszwecke in ihrer Auflösung vollautomatisiert angepasst werden können.

In Kapitel 4.1.1 wurde als Anforderung ein homogener Charakter des 3D-Modells aufgestellt. Durch einen einheitlichen Detailgrad sollte ein möglichst homogenes Erscheinungsbild des virtuellen, dreidimensionalen Campus realisiert werden. Der Grad der Detaillierung orientiert sich an den Anforderungen der verschiedenen Verwendungszwecke des Modells. Um ein bestimmtes Detaillierungsniveau einzuhalten, war es notwendig, einen entsprechenden Maßstab zu spezifizieren bzw. zu nutzen. Der so genannte Level-of-Detail-Ansatz (LoD) ist ein häufig genutztes Konzept, das Aufschluss über den Grad der Generalisierung bzw. der Detailliertheit eines dreidimensionalen Computermodells gibt. Bei der Erstellung dreidimensionaler Stadtmodelle großer Kommunen gilt dieses Konzept inzwischen als Quasi-Standard. Im Rahmen der Spezifizierung eines einheitlichen Datenmodells für 3D-Stadtmodelle wird ebenfalls dieser Entwurf herangezogen (OGC 2006, 11-12). Die Modellierung des Universitätscampus orientierte sich an diesem Konzept. Der LoD-Ansatz unterscheidet fünf verschiedene Detaillierungsgrade (Tab. 2). LoD 0 ist die am wenigsten detaillierte Stufe mit der stärksten Generalisierung. Ein 3D-Modell der Stufe LoD 0 enthält lediglich ein digitales Geländemodell, welches mit zweidimensionalen Daten überlagert wird, die beispielsweise Aufschluss über die Flächennutzung geben. LoD 4 ist die höchstdetaillierte Stufe mit der geringsten Generalisierung. Ein 3D-Modell der Stufe LoD 4 bietet begehbare Innenräume und bildet alle konstruktiven Elemente eines Gebäudes geometrisch ab. Die Lagegenauigkeit beträgt in dieser Stufe 0,2 m. Für das Vorhaben wurden die Stufen LoD 2 bzw. LoD 3 als Zielvorgabe gewählt. Die Ausgestaltung der Fassaden sollte aus oben genannten Gründen unter Verwendung von Texturen erfolgen, während die Grundfläche der Gebäude sich möglichst an der Realität orientieren bzw. sich konsistent zu den verwendeten 2D-Basisdaten verhalten sollte. Für einige Teile des Modells wurde aufgrund nicht zur Verfügung stehender oder noch nicht vorliegender Daten von diesen Detaillierungsstufen nach unten abgewichen. Dies gilt insbesondere für die zukünftigen Baustufen des Klinikrings sowie für die an das Untersu-

chungsgebiet angrenzende Bebauung. Für einige Objekte, insbesondere die Ein- und Zugangsbereiche von Kinderklinik, Medizinischer Klinik, Bioquant und Physikalischem Institut wurde der Detaillierunsgrad LoD 4 als Zielvorgabe gesetzt.

	LoD 0	**LoD 1**	**LoD 2**	**LoD 3**	**LoD 4**
Maßstabsebene	Regionalmodell	Stadtmodell; Klötzchen-modell	Stadtviertel; texturierte Modelle	Architekturmodell; geometrisch differenzierte Außenhülle	Architekturmo-dell; Innenräume begehbar
Detailgrad	Sehr gering	gering	mittel	hoch	Sehr hoch
Genauigkeit 2D/3D	Geringer als LoD 1	5m / 5m	2m / 2m	0,5m / 0,5m	0,2m / 0,2m
Generalisierung	Maximal (nur Flächennut-zung)	Objektblöcke > 6m*6m Grund-fläche in gene-ralisierter Form	Objekte > 4m*4m Grund-fläche in gene-ralisierter Form	Objekte > 2m*2m Grundfläche in realer Form	Reale Form; konstruktive Elemente und Öffnungen wer-den abgebildet
Dachform / Dachüberhang	- / -	Ebene Fläche / -	Dachtyp und Ausrichtung / -	Reale Dachform und Dachstruktur / -	Reale Dachform und Dachstruktur / ja

Tab. 2: Level of Detail Konzept (eigene Darstellung nach BACHMANN ET AL. 2003, 5 bzw. OGC 2006 11-12)

Das beschriebene LoD-Konzept trifft lediglich Aussagen über die Beschaffenheit der Geometriekomponente eines 3D-Modells. Qualität und Charakter der verwendeten Texturkomponenten werden nicht berücksichtigt. Um mit konkreten Zielvorgaben zu arbeiten, wurde diesbezüglich eine eigene Konzeption aufgestellt, welche als Ergänzung zum LoD-Entwurf verstanden werden kann. Hierzu wurden konkrete methodische Aspekte für die Erfassung und Verarbeitung der fotografischen Aufnahmen der Gebäudefassaden definiert, um auf diese Weise einen einheitlichen Detailgrad und homogenen Charakter für die Texturierung des Modells zu gewährleisten. Die bei der fotographischen Erfassung der Gebäudefassaden und anschließenden Verarbeitung der Aufnahmen berücksichtigten Hauptaspekte sind nachfolgend aufgeführt:

- Erfassung – Wahl der kleinmaßstäblichen Perspektive: die für die Texturierung notwendigen Fassaden lassen sich entweder aus der Luft, durch von einem Flugzeug aufgenommene Schrägbilder oder terrestrisch gewinnen. In der Literatur werden überwiegend finanzielle bzw. technische Gründe für die eine oder andere Methode genannt (STEIDLER 2004, 360 bzw. FRÜH & ZAKHOR 2003,

61). Für vorliegende Arbeit wurden der visuelle Eindruck sowie die Akzeptanz des Betrachters als maßgebend zu Grunde gelegt. Von der Luft aus aufgenommene Schrägbilder sind für 3D-Modelle, die auch für die Betrachtung aus Fußgängerperspektive gedacht sind, zum Teil wenig geeignet. Dies wird vor allem dann deutlich, wenn die Fassade aufgrund konstruktiver Elemente wie Balkone oder Außentreppen eine gewisse Tiefe aufweist. Die genannten Elemente werden bei einer Schrägbildaufnahme aus der Luft von oben aufgenommen. Dies ist für eine Betrachtung des Modells aus der Luft adäquat. Aus Fußgängerperspektive erwartet der Betrachter hingegen, die Unterseite bzw. bei größerer Entfernung ausschließlich die Vorderseite dieser Elemente zu sehen. Der visuelle Eindruck ist somit unstimmig und die Akzeptanz des Modells auf Nutzerseite gering. Da das in dieser Arbeit entwickelte Modell explizit auch für die interaktive Begehung konzipiert ist, wurde eine terrestrische Perspektive zur Aufnahme der Fassaden gewählt. Umgekehrt sind nämlich auf diese Weise gewonnene Fassadentexturen für eine Betrachtung des Modells aus der Luft gut geeignet. Durch den höheren Betrachterabstand wirkt die für eine Betrachtung von oben weniger stimmige Ansicht einer Fassade vergleichsweise nur gering störend.

- Erfassung – Wahl der großmaßstäblichen Perspektive: Ziel war es, möglichst unverzerrte und vollständige Ansichten der Fassaden zu erhalten. Dementsprechend wurde mit langen Brennweiten und daraus folgender großer Entfernung zur Fassade gearbeitet. Dadurch steht die Aufnahmeebene, in diesem Fall der elektronische Fotosensor, möglichst parallel zur Fassade, was eine geringe Verzerrung gewährleistet. Durch die hohe Entfernung kann die Fassade einerseits vollständig in einer Aufnahme erfasst werden, andererseits werden extreme perspektivische Ansichten oben genannter konstruktiver Fassadenelemente vermieden. In Bereichen des Untersuchungsgebiets, in denen die Bebauung dicht ist, konnte dieses Kriterium nicht ausreichend berücksichtigt werden, so dass umfangreiche Nachbearbeitungsmaßnahmen notwendig wurden.
- Erfassung – Wahl des Aufnahmezeitpunkts: Berücksichtigt man die Jahreszeit, so ist das Winterhalbjahr besser für die fotographische Erfassung geeignet. Bäume, die das Objekt verdecken, tragen kein Laub. Die Fassade kann dadurch vollständiger aufgenommen werden und der Aufwand der Nachbearbeitung ist vergleichsweise gering. Auch die Wetterverhältnisse sind zu dieser Jahreszeit

besser geeignet. Bedeckter Himmel und damit ausschließlich diffuse Beleuchtung des Objekts vermeiden harten Schattenwurf anderer Gebäude bzw. konstruktiver Elemente auf den Fassaden. Da für das Modell ein homogenes Erscheinungsbild und eine vielseitige Verwendbarkeit angestrebt wurden, mussten solche Schatten vermieden werden. Zu unterschiedlicher Tageszeit aufgenommene Fassaden können durch den unterschiedlichen, nicht stimmigen Schattenwurf den visuellen Eindruck beim Betrachter stören. Außerdem ist ein Modell, welches bereits in den Texturen einen Schatten enthält, nur noch bedingt für Verschattungsanalysen geeignet. Weiterhin wurde der Aufnahmezeitpunkt so gewählt, dass möglichst wenige störende Bildelemente wie Fußgänger oder Fahrzeuge die Sicht auf die Fassade behinderten.

- Verarbeitung – Entzerrung der Aufnahmen: Unter Berücksichtigung oben genannter Aspekte kann die perspektivisch unverzerrte Aufnahme einer Fassade nur im Idealfall erreicht werden. Der Großteil der Aufnahmen musste nachträglich manuell bearbeitet werden. Ziel war hierbei nicht eine photogrammetrische Entzerrung, die die exakte Vermessung von Fassadenelementen auf Basis des Fotos erlaubt, sondern eine perspektivische Entzerrung, die rechte Winkel bzw. Parallelen in der Fassade auch als solche abbildet. Dies wurde mit entsprechender Standardsoftware realisiert (Abb. 11).
- Verarbeitung – Bereinigung der Aufnahmen: Störende Bildelemente wie Fußgänger, Fahrzeuge und auch harte Schatten konnten nicht vollständig vermieden werden. Mittels nachträglicher Bearbeitung wurden diese Elemente aus den Aufnahmen entfernt. Um einen idealtypischen Charakter des Modells zu gewährleisten, wurden in diesem Arbeitsschritt auch zahlreiche Graffitis und Plakate von Fassaden getilgt. Die Entfernung solcher kurzlebigen Bestandteile trägt zu einem homogenen Gesamteindruck des Modells bei.
- Verarbeitung – Zusammensetzen der Aufnahmen: Viele Fassaden besitzen eine Ausdehnung, die eine vollständige Abbildung mit einer einzigen Aufnahme verhindern. In diesem Fall wurden mehrere Fotos gemacht, um eine vollständige Abdeckung zu erreichen. Der Einsatz von kurzen Brennweiten wurde vermieden, da sich gezeigt hat, dass der nachträglichen perspektivischen Entzerrung, die durch den Einsatz von Weitwinkelobjektiven begünstigt wird, Grenzen gesetzt sind. Voraussetzung für das Zusammensetzen einer Fassade aus mehreren Aufnahmen sind vollständig entzerrte Einzelaufnahmen.

In Kapitel 4.1.1 wurde die Einbettung des 3D-Modells in ein einheitliches räumliches Bezugssystem gefordert. Die 2D-Daten aus den unterschiedlichen Quellen wurden bereits im GIS nach Gauß-Krüger georeferenziert. Auch das 3D-Modell soll in dieses Raumbezugssystem eingebettet sein. Dem steht entgegen, dass in Geographischen Informationssystemen für die Abbildung großer Zahlen ein breiterer Wertebereich zur Verfügung steht als in Softwareumgebungen zur dreidimensionalen Modellierung. Um auch bei der dreidimensionalen Modellierung von Geoobjekten in einem georeferenzierten Bezugsraum arbeiten zu können, hat sich für das Arbeiten in entsprechenden Umgebungen die Einführung eines Offset bewährt (FREIWALD ET AL. 2006, 105). Dieses Offset wird beim späteren Export der Daten in das für das interaktive Modell verwendete Datenformat VRML übernommen, da eine vollständige Abbildung des Gauß-Krüger-Bezugsraums auch hier nur unzureichend möglich ist (FREIWALD & JANY 2005, 148). Das Offset sieht eine Koordinatentransformation um -3.470.000 m für den Rechtswert und -5.470.000 m für den Hochwert vor. Für Berechnungen oder Analysen wird das Offset entsprechend aufgeschlagen.

5.3.3 Geometrische Abbildung

In Kapitel 4.1.1 wurde eine Konsistenz zwischen 3D-Modell und zugrunde liegenden 2D-Daten postuliert. Die zur Modellierung der Gebäude notwendigen Grundrissdaten wurden hierzu über Standardaustauschformate unverändert aus dem Geographischen Informationssystem in die zur 3D-Modellierung ausgewählte Softwareumgebung 3ds max portiert (vgl. Abb. 11). Die vor dem Export durchgeführte, speziell an die Bedürfnisse für die 3D-Modellierung angepasste Georeferenzierung der Geometriedaten bleibt dabei vollständig erhalten. Auf den importierten Grundriss- bzw. Dachaufsichtplänen wurde unter Auswertung der in Kapitel 5.1 genannten Quellen die dreidimensionale Geometrie aufgebaut. Bei der Modellierung des Geländemodells wurde mit den dafür digitalisierten und erhobenen Daten entsprechend vorgegangen. Abbildung 13 illustriert in vereinfachter Weise die Ableitung und Basierung einer Gebäudegeometrie aus bzw. auf den Grundrissen. Die Abbildung zeigt ebenfalls einen Ausschnitt des realisierten Geländemodells mit dem auf einer Länge von etwa drei Kilometern digitalisierten Neckarbett sowie charakteristischen Abgrabungen im Bereich des Klinikrings. Spezielle Modellierungstechniken und Werkzeuge, die beim Aufbau der Geometrie zum Einsatz kamen, sind detailliert bei FREIWALD ET AL. (2006) beschrieben.

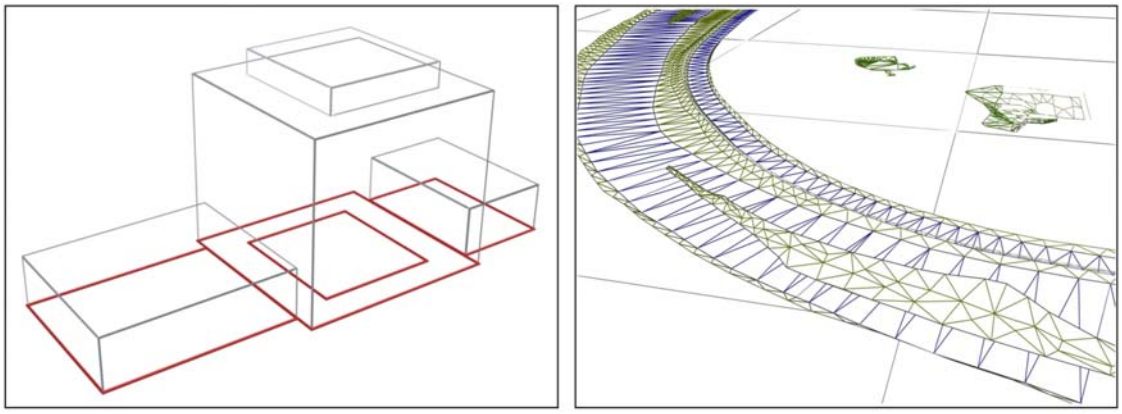

Abb. 13: links: Konsistenz zwischen zweidimensionalen Daten und 3D-Modell, Grundrissplan in rot und darauf basierendes 3D-Modell in schwarz als Drahtgitterdarstellung; rechts: Ausschnitt aus dem digitalisierten Geländemodell als Drahtgitterdarstellung (eigene Darstellung)

5.3.4 Texturen

Zur Texturierung der Geometrie wurden die zu diesem Zweck aufgenommenen Fassadenfotos analysiert und aufwendig bearbeitet (Abb. 11). Um die Auswertung und den Zugriff auf die insgesamt 1200 gemachten Aufnahmen effizient zu gestalten, wurden diese in eine Datenbank integriert. In einem ersten Schritt wurden die Fassadenfotos perspektivisch entzerrt (Abb. 14). Anschließend wurden die Fassaden extrahiert und optisch bereinigt (Abb. 15). Besonders breite Fassaden oder Gebäudeseiten, die aus anderen Gründen nicht mit einer einzigen Aufnahme abgebildet werden konnten, wurden aus mehreren Fotos zusammengesetzt. Danach wurden in der zur dreidimensionalen Modellierung gewählten Softwareumgebung 3ds max Texturbibliotheken aufgebaut und die einzelnen Texturen gezielt den entsprechenden Gebäudegeometrien zugewiesen (Abb. 11). Für die rund 100 Bauten des Untersuchungsgebiets fanden insgesamt 350 verschiedene Fassadentexturen im Modell Verwendung. Bei standardisierten Typenbauten wurden einzelne Texturen mehrfach verwendet. Eine Zuweisung von Texturen nach dem Zufallsprinzip, wie sie beispielsweise in großem Umfang bei der Generierung des 3D-Stadtmodells von Berlin angewandt wurde (BERLIN PARTNER GMBH 2006), wurde in dieser Arbeit strikt vermieden, da dies dem Anspruch einer vollständigen und realitätsgetreuen Abbildung des Untersuchungsgebiets widersprochen hätte.

Zur Generierung der Fassaden für nicht existente Bauten zukünftiger oder historischer Planung waren zusätzliche Schritte notwendig. Um die Fassaden der nicht realisierten Klinikbauten des historischen Schmiederplans detailliert abbilden zu können, wurden synthetisch Texturen angefertigt. Diese basieren auf den Aufrissplänen von 1933 und dem historischen Holzmodell (vgl. Kap. 5.1.2). Sie orientieren sich zudem am heutigen Aussehen der Fassaden der Chirurgischen Klinik, die als einziges Gebäude des Generalbebauungsplans von 1932 realisiert wurde. Für die Gebäude der zukünftigen Planung wurde analog verfahren. Zusätzlich wurden hier die Projektpläne beauftragter Architekten ausgewertet. Spezielle Techniken und Werkzeuge, die bei der Texturierung zum Einsatz kamen, sind detailliert bei FREIWALD ET AL. (2006) beschrieben.

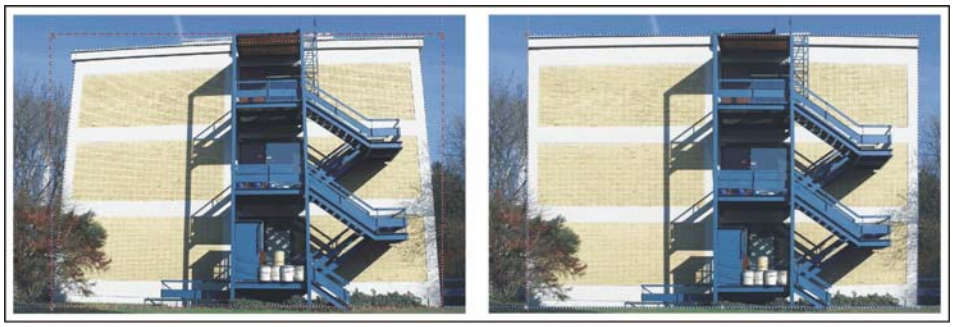

Abb. 14: Perspektivische Entzerrung der Fassadenaufnahmen am Beispiel der Westfassade des Chemischen Instituts INF 273. links: Weitwinkelaufnahme aus großer Nähe bei niedrigem Aufnahmepunkt mit daraus resultierender Verzerrung; rechts: nach manueller perspektivischer Entzerrung (eigene Aufnahme und Darstellung)

Abb. 15: Optische Bereinigung der Fassadenaufnahmen am Beispiel der Ostansicht des Mineralogischen Instituts INF 236. links: bereits perspektivisch korrigierte Ansicht mit störenden, nicht zur Fassade gehörenden Bildelementen; rechts: manuell vollständig bereinigte Ansicht (eigene Aufnahme und Darstellung)

5.3.5 Ausgewählte Ergebnisse

Die Realitätstreue und hohe Qualität des in dieser Arbeit geschaffenen 3D-Modells des Baubestands kann jederzeit vor Ort verifiziert werden. Um aber die Qualität und die Zuverlässigkeit der Abbildung zukünftiger Bauvorhaben mithilfe des Konzepts dreidimensionaler Visualisierung demonstrieren bzw. verifizieren zu können, wurden zu Beginn der Arbeit zwei sich im konkreten Planungsstadium befindliche Bauprojekte ausgewählt, deren Realisierung bis zum Abschluss vorliegender Arbeit zu erwarten war. Dies sind zum einen das Bioquantgebäude, welches im März 2007 fertig gestellt wurde, sowie die neue Kinderklinik, die noch 2007 bezogen werden soll. Naturgemäß ist es nicht möglich, Fassaden noch nicht bestehender Bauten fotographisch zu erfassen. Um dennoch eine möglichst realistische Modellierung der genannten Gebäude zu erreichen, wurden detaillierte Planwerke der mit der Umsetzung beauftragen Architekten ausgewertet. Während die Aufrisspläne des Bioquantgebäudes texturartige Elemente enthielten, die Eingang in die Modellierung fanden, wurden für Teile des 3D-Modells der Kinderklinik einerseits synthetische Texturen angefertigt und andererseits eine voll ausdifferenzierte Detailmodellierung umgesetzt, die eine Begehbarkeit des Gebäudes möglich macht.

Abb. 16: links: Anfang 2005 prognostiziertes Erscheinungsbild des Bioquantgebäudes im 3D-Modell; rechts: tatsächliches Aussehen bei Fertigstellung Anfang 2007 (eigene Darstellung bzw. Aufnahme)

Die Abbildungen 16 und 17 stellen jeweils das auf Basis der vierdimensionalen Modellierung bereits Anfang 2005 prognostizierte Erscheinungsbild dem tatsächlichen, realen Aussehen der Gebäude inklusive ihrer baulichen Umgebung Anfang 2007 ge-

genüber. Eine nachträgliche Anpassung des Modells an das tatsächliche Erscheinungsbild wurde bewusst nicht vorgenommen, um eine Bewertung der Prognosequalität nicht zu verfälschen.

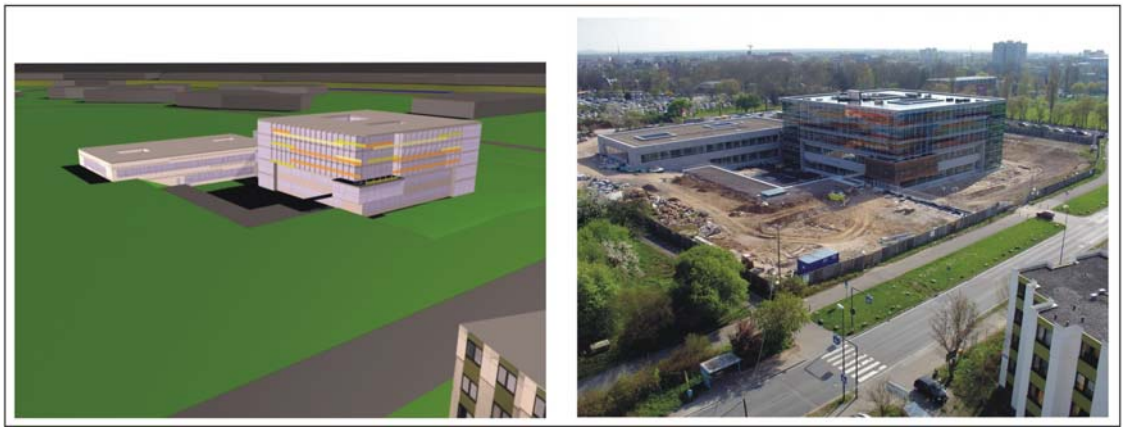

Abb. 17: links: Anfang 2005 prognostiziertes Erscheinungsbild der Kinderklinik im 3D-Modell; rechts: tatsächliches Aussehen bei Fertigstellung Anfang 2007 (eigene Darstellung bzw. Aufnahme)

Die Abbildungen 18 bis 22 stellen weitere, ausgewählte Ausschnitte und Ergebnisse aus dem dreidimensionalen Modell des Untersuchungsgebiets dar. Weitere Ergebnisse, die das vollständige 3D-Modell des Heidelberger Universitätscampus im Kontext mit unterschiedlichen Informationsebenen visualisieren, finden sich in Form eines Films auf im Anhang beigefügter CD.

Abb. 18: links: Bereich des zukünftigen Klinikums als Bestand 2007; rechts: die zukünftige Planung für denselben Bereich bis zum Jahr 2020 unterlegt mit der Gesamtplanung des Universitätsbauamts; das lokale Geländemodell um die Kliniken wird in beiden Abbildungen durch eine netzartige Darstellung betont (eigene Darstellung)

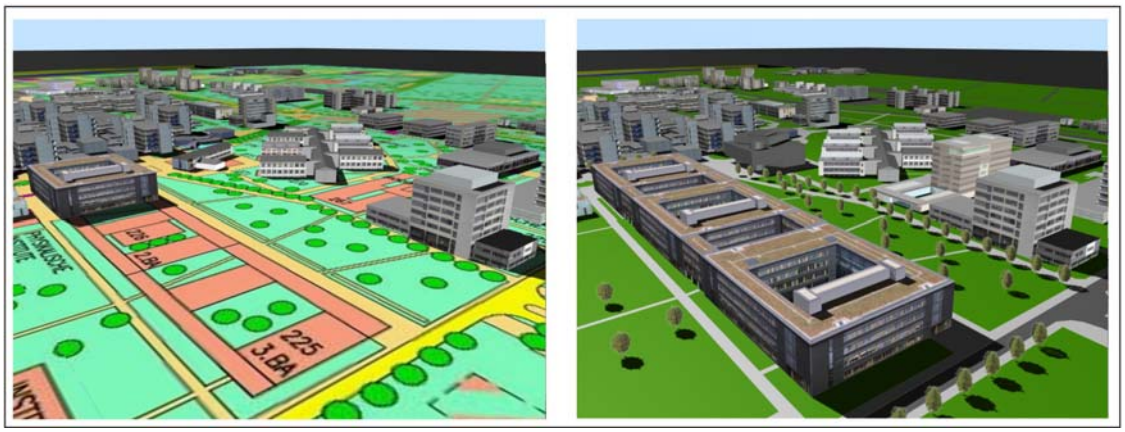

Abb. 19: links: Bereich des zentralen Forums als Bestand 2005 unterlegt mit der Gesamtplanung des Universitätsbauamts; rechts: die zukünftige Planung für denselben Bereich bis zum Jahr 2020 (eigene Darstellung)

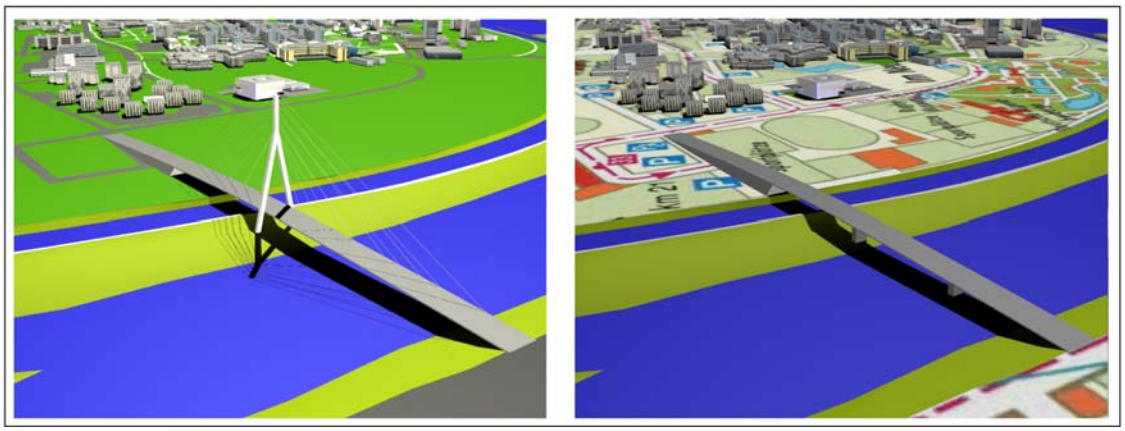

Abb. 20: Bereich für die diskutierte fünfte Neckarquerung in der westlichen Verlängerung des Klausenpfades. links: Variante Schrägseilbrücke; rechts: Variante Balkenbrücke unterlegt mit dem Stadtplan Heidelberg (eigene Darstellung)

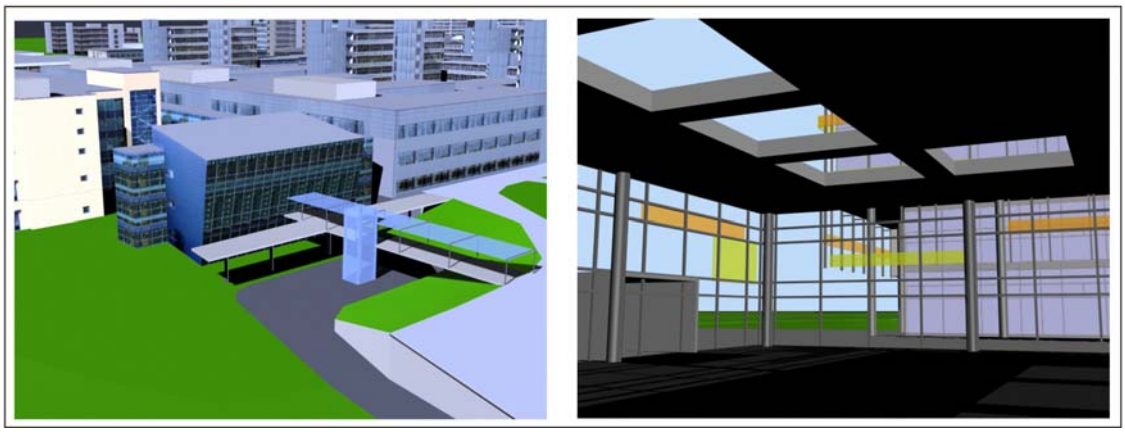

Abb. 21: links: Detailansicht des Zugangsbereichs der Medizinischen Klinik INF 410; rechts: Innenansicht des begehbar modellierten Foyers der Kinderklinik INF 430, welche Mitte 2007 fertig gestellt wird (eigene Darstellung)

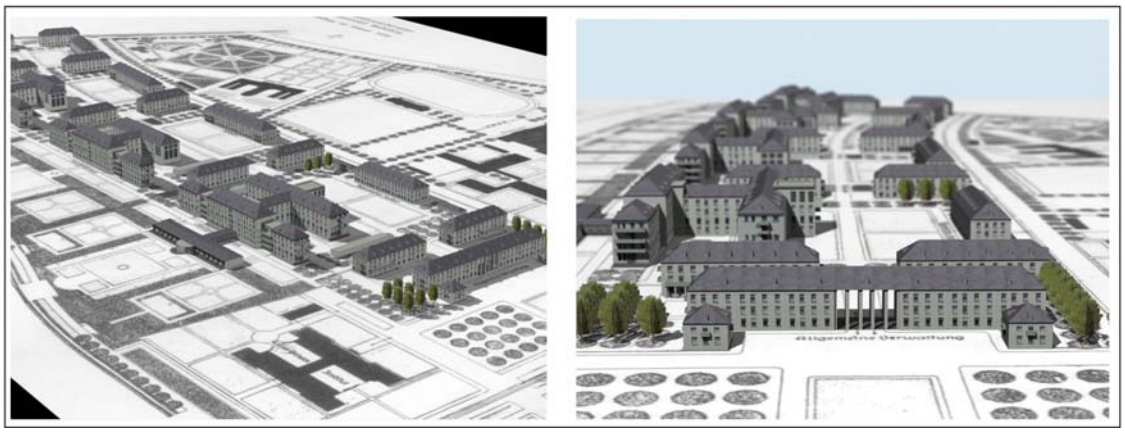

Abb. 22: Ansichten der nicht realisierten Planung für die Verlegung sämtlicher Kliniken von 1932 unterlegt mit dem Entwurf von Schmieder (eigene Darstellung)

6 Implementierung des Informationssystems

Auf Grundlage der in Kapitel 4.2 aufgestellten Anforderungen an eine interaktive, webbasierte Informations- und Planungsanwendung wurde vom Verfasser ein Prototyp entwickelt. Dessen Online-Funktionstauglichkeit wurde erfolgreich getestet (vgl. CD im Anhang).
Die Implementierung wurde mit den in Kapitel 6.1 beschriebenen Programmiertechniken und -sprachen realisiert.
Da es eines der erklärten Ziele des Konzeptes ist, die Architektur gegenüber Erweiterungen durch Dritte möglichst offen zu gestalten, wurden hinsichtlich der Datenhaltung nur standardisierte Formate verwendet. Dasselbe gilt für die Implementierung der Funktionalitäten des Informationssystems. Auch hier kamen nur Programmiersprachen bzw. -konzepte zur Anwendung deren Dokumentation jedem offen zugänglich und deren Weiterentwicklung größtenteils unabhängig von kommerziellen Interessen einzelner Konzerne ist. Auf diese Weise ist sicher gestellt, dass quasi jeder das System nach seinen Vorstellungen erweitern kann; inhaltlich wie funktional. Darüber hinaus werden durch die Verwendung standardisierter Techniken und Formate Abhängigkeiten von bestimmten Herstellern vermieden und gleichzeitig die Investitionssicherheit des Informationssystems gewährleistet.

6.1 Technische Grundlagen

Die Auswahl der Techniken, mit denen die Online-Plattform realisiert wurde, orientiert sich an den im Kapitel 4.2 aufgestellten Anforderungen. Die Verwendung von standardisierten Technologien, deren Spezifikation jedem frei zugänglich ist und die von nichtkommerziellen Organisationen weiterentwickelt werden bzw. die frei von Patentansprüchen sind, hatte dabei die oberste Priorität. Der Großteil der eingesetzten Programmiersprachen und Konzepte basiert auf Standards des World Wide Web Consortiums, kurz W3C. Dies ist ein anerkanntes Gremium für die Standardisierung das World Wide Web betreffender Techniken. Die offizielle Klassifizierung als Standard

kommt der International Organization for Standardization, kurz ISO, zu. Viele der Techniken, die das W3C entwickelt hat, gelten als de-facto Standard. Die am weitesten verbreiteten Techniken, wie beispielsweise HTML, wurden inzwischen auch von der ISO als offizielle Standards spezifiziert.

Es ist nicht Ziel, eine umfassende Abhandlung des Mediums Internet oder aller damit verbundenen Programmiertechniken zu liefern. Daher wird lediglich auf diejenigen Technologien und Programmiersprachen eingegangen, die für die Entwicklung des prototypischen Informationssystems wesentlich sind. Neben einem allgemeinen Überblick über die jeweilige Technologie werden diejenigen Elemente, die in Kapitel 6.5 eine besondere Rolle spielen, näher beschrieben.

6.1.1 Internet – Dienste und Struktur

Das Internet ist ein Netz von weltweit verteilten Rechnern. Dieses Netz kann unterteilt werden in verschiedene weitere Netze. Die Kommunikation aller Rechner findet auf Basis des standardisierten TCP-Protokolls (Transmission Control Protocol) statt. Dieses einheitliche Kommunikationsprotokoll ermöglicht es, dass auch Rechner unterschiedlicher Betriebssysteme miteinander kommunizieren können (LINKE UND WINKLER 1999, S. 364).
Das Internet bietet verschiedene so genannte Dienste. Die bekanntesten sind Email, File Transfer Protocol (FTP), Newsgroups und World Wide Web (WWW). Email erlaubt den Austausch persönlicher elektronischer Nachrichten; dies ist der häufigst genutzte Internetdienst. Über den FTP-Dienst kann der Nutzer eine Verbindung mit einem Server aufbauen, um von diesem Daten auf seinen Rechner herunterzuladen oder umgekehrt. Eine Newsgroup ist eine Art virtuelles Schwarzes Brett. Hier können zu bestimmten Themenbereichen Nachrichten oder Beiträge, so genannte „postings“, hinterlassen, „geposted“, werden. Nachrichten zu einem speziellen Thema werden dabei in so genannten „Threads“ zusammengefasst. Wenn man vom „Internet“ spricht, meint man damit meist dessen Dienst World Wide Web. Das WWW ist ein Hypertext-System, in dem Textdokumente und andere Arten von Dokumenten miteinander durch so genannte Hyperlinks verbunden sind. Die technische Grundlage des WWW bildet die standardisierte Auszeichnungssprache HTML (Hypertext Markup Language) (MEINEL UND SACK 2004, 93 ff.).

Bei den Diensten des Internets geht es um den Austausch von Daten. Das Grundgerüst für diesen Datenaustausch bildet die so genannte Client-Server-Architektur. Hierbei werden Ressourcen von einem zentralen Computer (Server) vorgehalten, auf die ein Benutzer mit seinem Arbeitsrechner (Client) zugreifen kann. Der Zugriff findet in Form einer Anfrage des Clients statt. Diese Anfrage wird vom Server mit der Übergabe der angefragten Daten beantwortet. Unter Client bzw. Server wird nicht nur die Hardware, also die physische Ausprägung der Rechner, verstanden, sondern auch die für den Datenaustausch notwendigen Softwarekomponenten. Im Zusammenhang mit der Nutzung des WWW ist dies auf Clientseite üblicherweise ein Web-Browser (MÜNZ 2007). Abbildung 23 stellt in schematischer Art und Weise die klassische Client-Server-Architektur einer Web-Anwendung dar.

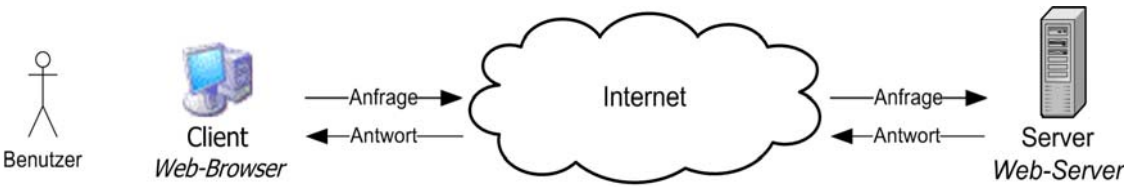

Abb. 23: Client-Server-Architektur (eigene Darstellung)

Damit ein Server innerhalb des Internets gezielt aufgerufen werden kann, muss dieser mit einer eindeutigen Adresse ausgestattet sein. Um einen Rechner im WWW anzusprechen, rufen Web-Browser diesen über den so genannten URL (Uniform Resource Locator) auf. Der URL ist eine genormte Art der Adressierung von Rechnern. Er besteht aus mehreren Teilen nach folgendem Schema (LINKE & WINKLER 1999, 756 ff.):

Protokoll://Dienst.Rechnername.Domain/Verzeichnis/Dokument

Die einzelnen Abschnitte eines URL sind wie folgt definiert (KERSKEN 2003, 680 ff.): Als Protokoll wird zur Adressierung im World Wide Web das HTTP-Protokoll (HyperText Transfer Protocol) eingesetzt. Dies ist Teil der eingangs erwähnten TCP-Protokoll-Familie. Als Dienst wird das WWW in Anspruch genommen. Der Rechnername ist theoretisch ein vom Betreiber des Rechners frei wählbarer Name, der allerdings gewissen Konventionen genügen muss und in Abhängigkeit des Dienstes und der Domain nur einmal vergeben werden kann. Der Rechnername wird im Internet über den Domain Name Service in eine so genannte IP-Adresse (Internet Protocol) umgewandelt (KOLKMANN 2005). Die Domain kennzeichnet meist das Land, in dem

der Rechnername vergeben wurde. Der Verzeichnis-Teil eines URL kann eine Pfadangabe innerhalb einer Ordnerstruktur enthalten, um einen bestimmten Ordner zu identifizieren. Der letzte Teil des URL bezeichnet das Dokument, welches aufgerufen werden soll.

6.1.2 HyperText Markup Language – HTML

Ursprünglich war HTML als so genannte Auszeichnungssprache für die reine Gliederung und Beschreibung von Dokumenten entworfen worden. Mit deren Hilfe sollte der Austausch von Dokumenten zwischen verschiedenen Plattformen ermöglicht werden. HTML basiert auf der von der International Organization of Standardization (ISO) normierten Meta-Sprache SGML (Structured Generalized Markup Language), mit deren Hilfe Auszeichnungssprachen für bestimmte Dokumenttypen definiert werden (ISO 1986; BERNERS-LEE & CONNOLY 1995). Eine ausführliche Beschreibung der Metasprache SGML ist bei GOLDFARB (1993) zu finden.
Eine Teilmenge der HTML-Spezifikation wurde 2000 von der ISO als Standard spezifiziert (ISO 2000). HTML gilt heute als Lingua Franca des Internets. Seit dem ersten Entwurf hat sie sich von der eigentlichen Zweckbestimmung, der gliedernden Beschreibung von Dokumenten, hin zu einer Sprache entwickelt, die auch das äußere Erscheinungsbild eines Dokuments definiert und multimediale Elemente zur Erweiterung des Dokuments einbindet (MÜNZ 2007; RAGGETT, LE HORS & JACOBS 1999).

Die Darstellung von HTML-Dokumenten erfolgt mit Hilfe von Web-Browsern. Das HTML-Dokument wird dabei vom Browser geladen, die einzelnen Tags bzw. Elemente werden interpretiert und der Inhalt des Dokuments auf dem Monitor angezeigt. Da die gültigen HTML-Spezifikationen nicht von allen Browser-Herstellern auf die gleiche Arte und Weise umgesetzt werden, kommt es vor, dass ein HTML-Dokument in verschiedenen Web-Browsern unterschiedlich dargestellt wird. Aktuelle Browser decken aber zunehmend größere Teile der HTML-Spezifikation ab, so dass der Aufwand, den Autoren und Entwickler betreiben müssen, ein HTML-Dokument so zu gestalten, dass es mit verschiedenen Web-Browsern gleich dargestellt wird, geringer wird (BRAUN 2007, 162 ff.). Als Entwicklungsbasis für das Informationssystem wurden das Betriebssystem Microsoft Windows XP und der Web-Browser Internet-Explorer derselben Firma ausgewählt. Windows XP ist mit über 75% Anteil das mit Abstand am häufigsten eingesetzte Betriebssystem. Dasselbe gilt für die Verbreitung des Internet-

Explorers, der einen Anteil von rund 60% an allen eingesetzten Web-Browsern besitzt (W3SCHOOLS 2007a).

Der Quelltext von HTML-Dokumenten wird als so genannter Klartext im ASCII-Format abgespeichert. ASCII ist ein ISO-Standard von 1967, der den Austausch von Textdaten definiert (ISO 1997). Das bedeutet, dass in HTML geschriebene Dokumente mit jedem beliebigen Texteditor geöffnet und verändert werden können (LINKE & WINKLER 1999, 339).
Neben dem eigentlichen Inhalt, der mittels HTML-Code transportiert werden soll und bei dem es sich ursprünglich um reinen Text handelte, besteht ein HTML-Dokument aus Befehlen, die das Dokument gliedern und die Darstellung der Inhalte bestimmen. Diese Befehle werden auch Tags genannt und innerhalb des Dokuments in spitzen Klammern notiert. Tags umschließen Bereiche innerhalb eines Dokuments, deren Darstellung sie definieren. Diese von Tags eingeschlossenen Bereiche werden auch Elemente genannt. Das Aussehen oder weitergehende Eigenschaften dieser Elemente wird über Attribute und Parameter bestimmt, die dem jeweiligen Tag zugeordnet werden können. Um Elemente innerhalb des Dokuments anzusprechen, können sie mit einer eindeutigen ID ausgestattet werden. Mit Hilfe dieser ID ist auch der Zugriff von außerhalb des Dokuments möglich (CASTRO 2000; 21 ff.).

HTML-Dokumente besitzen eine hierarchische Struktur. Auf der hierarchisch obersten Ebene stehen zwei Elemente; der HTML-Kopf und der HTML-Körper. Im Dokument selbst werden diese Elemente mit den Tags <head> bzw. <body> gekennzeichnet.
Der Kopf enthält in entsprechenden Tags Informationen zum Titel des Dokuments sowie Meta-Informationen wie Autor, letzte Aktualisierung und dergleichen. Über das Element <style> können an zentraler Stelle so genannte Stylesheets definiert werden. Hierbei handelt es sich um eine Art von Formatvorlage, nach der die Darstellung ihr zugewiesener Elemente vorgenommen wird. Stylesheets können auch als separate CSS-Dateien (Cascading Style Sheets) ausgelagert werden (ANDREW & SHAFER 2006). Auf die Eigenschaften von CSS wird im nachfolgenden Kapitel detaillierter eingegangen.
Das Element <script> des HTML-Kopfes stellt einen wesentlichen Bestandteil für die Erweiterung der Funktionalität eines HTML-Dokuments dar. Es erlaubt die Einbindung oder externe Referenzierung von Script-Programmen.

Der HTML-Body enthält diejenigen Elemente, die den sichtbaren Inhalt einer Webseite beschreiben. Während die Zahl der HTML-Elemente, die im HTML-Kopf enthalten sein können, überschaubar ist, so würde eine erschöpfende Darstellung der Möglichkeiten des Bodys an dieser Stelle zu weit führen. Nachfolgend werden deshalb lediglich einige der HTML-Elemente knapp erläutert, die in der Implementierung der Online-Plattform eine Rolle spielen. Die Ausführungen basieren auf der Online-Referenz SELFHTML (MÜNZ 2007) und der offiziellen HTML-Spezifikation (RAGGETT, LE HORS & JACOBS 1999). Umfassende Abhandlungen des Themas sind auch bei BALZERT (2003), BORN (2000) und HIRSEMANN (2003) zu finden.

- ** (division)**: DIV-Elemente markieren einen allgemeinen Bereich innerhalb des HTML-Dokuments. Man kann diesen Tag auch als eine Art Gruppierung von verschiedenen weiteren Elementen auffassen. Innerhalb des DIV-Elementes können beispielsweise Texte, Grafiken oder Bilder eingeschlossen werden. Über das Attribut style kann das DIV-Element mit all seinen Inhalten unter Anwendung sämtlicher CSS-Eigenschaften (vgl. Kap. 6.1.3) formatiert werden. So kann zum Beispiel die Position von Inhalten innerhalb einer Webseite bestimmt werden.
- **<table>**: Mithilfe dieses Elements werden Tabellen definiert. Diese sind ursprünglich dazu gedacht, Daten in tabellarischer Form darzustellen. Tabellen können jedoch auch als gestalterisches Mittel genutzt werden, um beispielsweise Grafiken und Textabschnitte genau auf dem Bildschirm zu positionieren.
- **<form> (formular)**: Über Formulare kann der Benutzer Eingabefelder mit Text füllen, Einträge aus Listen wählen oder Schaltflächen aktivieren. Die vom Benutzer eingegebenen Inhalte werden nach Eingabe an den Server gesendet und dort ausgewertet. Die Anwendungsmöglichkeiten sind sehr breit.
- **<select>**: Mit dem SELECT-Element können innerhalb von Formularen Auswahllisten erzeugt werden. Diese können eine beliebige Anzahl von Einträgen enthalten, die der der Benutzer auswählen kann.
- **<option>**: Mit diesem Tag werden die einzelnen Einträge der SELECT-Elemente definiert. OPTION-Elemente übertragen über den ihnen hierarchisch nachgeordneten Tag <value> den darin abgelegten Wert an den Server. Statt

eines Wertes kann im Tag <value> aber auch der Aufruf eines Script-Programms gestartet werden.

- **<input>**: Das INPUT-Element definiert konkrete Bereiche bzw. Elemente, in denen der Benutzer Eingaben tätigen kann. Mit Hilfe des INPUT-Elements werden beispielsweise so genannte Checkboxen erzeugt. Dabei handelt es sich um kleine Quadrate, die der Benutzer durch Anklicken ankreuzen kann. In Abhängigkeit davon, ob eine solche Checkbox angekreuzt ist oder nicht, werden mit ihr verbundene Einträge an den Server geschickt. Über Event-Handler (vgl. Kap. 6.1.4), die mit den Checkboxen verbunden sind, können Script-Programme aufgerufen werden. Über das INPUT-Element werden auch Eingabefelder erzeugt, in die der Benutzer einen beliebigen Text oder beispielsweise einen URL eintragen kann, der dann vom Server ausgewertet wird. Diese Eingabefelder können auch zur Ausgabe von Text oder Zahlenwerten genutzt werden. Dann wird der Schreibzugriff über ein spezielles Attribut unterbunden.
- **<a> (anchor)**: Das ANCHOR-Element enthält Verweise, die auf andere Dokumente oder Dateien zeigen. Über diese Verweise, die auch Hyperreferenzen genannt werden, werden andere Dokumente aufgerufen oder in das aktuelle Dokument eingebunden. Dies stellt den grundlegenden Mehrwert jeder Webseite dar; nämlich die Verknüpfung von Dokumenten verschiedenen Inhalts.
- **<img> (image)**: Mit dem IMG-Element werden Grafiken oder Bilder in eine Webseite eingebunden. Über das Attribut src (source) wird die einzubindende Datei referenziert.
- **<object>**: Das OBJECT-Element ermöglicht das Einbinden der verschiedensten Datentypen; auch von solchen, deren Darstellung der Browser mit seinen Standardkomponenten zunächst nicht unterstützt. Dies können zum Beispiel Excel-Dateien, Musik- oder Film-Dateien, Flash-Animationen oder auch 3D-Daten und nahezu jede andere Art von Multimedia-Datei sein. Über entsprechende Attribute und Parameter innerhalb des OBJECT-Elements wird definiert, welche Art von Daten dargestellt werden sollen. Sofern diese keine native Unterstützung durch den Browser besitzen, wird dem Browser mitgeteilt, welche Erweiterungen er zur Darstellung benötigt und wo diese verfügbar sind. Im Idealfall läuft die Installation dieser Erweiterungen vollautomatisch und ohne Zutun des Benutzers ab.

6.1.3 Cascading Style Sheets – CSS

CSS ist eine Sprache mit der die Formateigenschaften von HTML-Elementen definiert werden. Sie ist somit eine direkte Ergänzung von HTML, um die Darstellungsmöglichkeiten von Web-Inhalten zu erweitern. Mit CSS können Schriftarten und -größen, Farben einzelner Elemente, deren Position innerhalb einer Webseite sowie eine Vielzahl an weiteren Formateigenschaften definiert und beeinflusst werden. CSS wird wie HTML auch vom W3C weiterentwickelt und gilt als de facto Standard zur Formatierung und Darstellung von HTML-basierten Inhalten (MÜNZ 2007).
CSS kann auf verschiedene Arten in HTML-Dokumenten verwendet werden. Eine häufig angewandte Methode ist die Deklaration externer CSS-Dateien im Kopf der HTML-Datei über das Element <style> (ANDREW & SHAFER 2006, 41 ff.). Bestimmte Inhalte des HTML-Dokuments werden dabei auf eine spezielle Art ausgezeichnet und einer in der CSS-Datei definierten Formatvorlage zugewiesen. Man spricht in diesem Fall auch von externen Stylesheets. Eine andere Möglichkeit der Verwendung von CSS ist das Einbinden eines speziellen CSS-Attributes <style> direkt in einzelnen HTML-Tags (RAGGETT 2002).
Eine umfassende, aktuelle Abhandlung des Themas bietet MEYER (2006).

In der Realisierung der Online-Plattform wird CSS insbesondere für die Positionierung einzelner Elemente innerhalb der graphischen Benutzeroberfläche eingesetzt. Auch bei der Gestaltung der interaktiven Schaltflächen des Informationssystems spielt CSS eine wesentliche Rolle.

6.1.4 JavaScript

JavaScript wurde 1995 vom Browserhersteller Netscape eingeführt, um die Möglichkeiten bei der Architektur von Webseiten mit aktiven Inhalten zu erweitern. Unter dem Namen ECMAScript wurde JavaScript 1999 von der Standardisierungs-Organisation ECMA (European Computer Manufacturers Association) normiert (ECMA 1997). Seit 1998 ist JavaScript auch nach ISO standardisiert (ISO 1998).
Seit der zweiten Hälfte der 1990er Jahre obliegt die Weiterentwicklung von JavaScript dem W3C. Das W3-Consortium erarbeitet bzw. pflegt jedoch keine konkrete Spezifikation für JavaScript, sondern definiert ein allgemeines Modell aller in einem Dokument möglichen Elemente, an welches JavaScript seitdem angepasst wurde bzw. wird

(MÜNZ 2007). Bei diesem Modell handelt es sich um das Document Object Model (DOM), welches im nachfolgenden Kapitel behandelt wird.

Im Gegensatz zu anderen Programmiersprachen werden Script-Sprachen wie JavaScript nicht kompiliert, d.h. in einen für den Computer direkt ausführbaren Maschinencode übersetzt. Dies macht die Ausführung von JavaScript-Code vergleichsweise langsam, was sich allerdings erst bei sehr großen rechenintensiven Projekten negativ auswirkt. Andererseits bietet der Wegfall der Kompilierung erhebliche Vorteile bei der Entwicklungsdauer von Softwareapplikationen, da der Script-Code direkt und ohne vorheriges Übersetzen in Maschinencode getestet werden kann (PREIMESBERGER 2003). Auch Script-Sprachen müssen in Maschinensprache übersetzt werden, damit der Rechner diese ausführen kann. Dieses Übersetzen, man spricht auch von Interpretieren, wird aber erst durchgeführt, wenn der Quelltext während des Programmablaufs benötigt wird. Bei klassischen Programmiersprachen wird der Quelltext bereits vorher übersetzt. Der kompilierte Code wird als neue Datei in Maschinensprache abgespeichert. Der Geschwindigkeitsvorteil klassischer Programmiersprachen resultiert auch aus dieser Zeitersparnis heraus (SCHWICHTENBERG ET AL. 2004, 18). Einen empirischen Vergleich von Script- und Nonscript-Sprachen liefert PRECHELT (2000).
Im Unterschied zu anderen Script-Sprachen wie PHP oder Perl ist JavaScript eine clientseitige Scriptsprache, d.h. die Ausführung des Codes findet auf dem Clientrechner statt. Dies ist vorteilhaft, wenn auf Interaktionen des Benutzers möglichst schnell reagiert werden soll und dazu keine neue Daten vom Server benötigt werden. Darüber hinaus ist so auch der von einem Server unabhängige Einsatz der Software-Applikation möglich; vorausgesetzt, dass die notwendige Datengrundlage lokal verfügbar gemacht wird.
JavaScript ist nicht wie häufig angenommen eine vereinfachte Version der Programmiersprache Java. Ganz im Gegenteil ist JavaScript eine vollwertige, objektorientierte Programmiersprache, deren Komplexität anderen Programmiersprachen nicht nachsteht (FLANAGAN 2002, 2). Die Namensähnlichkeit beruht auf Marketingüberlegung, da Java einige Zeit vor JavaScript auf den Markt kam und sich bei der Einführung von JavaScript bereits erfolgreich etabliert hatte. JavaScript sollte ursprünglich LiveScript heißen. Der Name wurde erst kurz vor der Einführung geändert (KERSKEN 2003, 2007 ff.). JavaScript und Java besitzen bezüglich ihrer Syntax Gemeinsamkeiten, da sie beide auf der Tradition der Programmiersprache C aufbauen. Beide Sprachen sind dafür geeignet, ausführbare Inhalte in Web-Browsern bereitzustellen, wobei JavaScript

wesentlich mehr Möglichkeiten bietet, das Verhalten und den Inhalt von Browsern zu steuern. Während der Code von JavaScript über einen entsprechenden Interpreter direkt im Browser ausgeführt wird, muss für den Ablauf von Java-Programmen eine spezielle Laufzeit-Umgebung, die Java Virtual Machine (JVM), zur Verfügung stehen und gegebenenfalls gesondert durch den Benutzer installiert werden (SEEBOERGER-WEICHSELBAUM 2001, 28). Da unterschiedliche Web-Browser jedoch teilweise verschiedene Laufzeitumgebungen für Java verwenden und darüber hinaus neben der JVM der Firma Sun auch eine JVM der Firma Microsoft existiert, kann es zu Inkompatibilitäten beim Einsatz von Java kommen (MICROSOFT 2007; SUN 2007; STEPPAN 2003, 256). Java liegt zudem im Gegensatz zu JavaScript keine internationale Standardisierung zugrunde.
Die aufgezeigten Eigenschaften von JavaScript unterstützen die im Kapitel 4.2.3 aufgestellten Anforderungen an die technischen Grundlagen der Online-Plattform von allen in Betracht gezogenen Programmiersprachen am besten. An dieser Stelle soll keine Einführung in die Programmiersprache JavaScript geleistet werden. Im Folgenden werden daher nur einige Aspekte erläutert, die an späterer Stelle wieder aufgegriffen werden. Eine umfassende, aktuelle Einführung in JavaScript liefern FLANAGAN (2006) und KOCH (2006).

Die ereignisbasierte Programmierung ist ein Paradigma der modernen Softwareentwicklung, bei dem der Programmablauf durch direkt oder indirekt vom Benutzer ausgelöste Ereignisse gesteuert wird. Im Gegensatz dazu steht das Modell der stapelbasierten Programmierung, bei dem der Ablauf eines Programms durch den Programmierer fest vorgegeben ist. Software-Applikationen, die dem Benutzer eine Interaktion gestatten, basieren üblicherweise auf erstgenanntem Modell (FERG 2006, 22). Eine Übersicht dieser und weiterer Paradigmen der Softwareentwicklung liefern GRABMÜLLER (2003) bzw. JACKSON (1979). Der Entwicklung des interaktiven Informationssystems für den Heidelberger Universitätscampus wurde das Paradigma der ereignisbasierten Programmierung zu Grunde gelegt.
Um solche Ereignisse zu erkennen und den Programmablauf in Abhängigkeit davon steuern zu können, sind bestimmte Ereignisbehandlungsroutinen notwendig. Wesentlicher Bestandteil dieser Routinen sind die so genannten Event-Handler. Von diesen steht in JavaScript eine ganze Reihe zur Verfügung. Event-Handler können direkt in HTML-Elementen eingefügt werden und stellen ein wichtiges Bindeglied zwischen HTML und JavaScript dar. Häufig angewandte Event-Handler sind onClick oder

onChange. Verbindet man den Event-Handler onClick beispielsweise mit einer Schaltfläche, so registriert er, wenn der Benutzer mit der Maus auf diese Schaltfläche klickt. Wenn Event-Handler ein Ereignis registrieren, führen sie definierbare Anweisungen aus. So können bei bestimmten Benutzerinteraktionen spezifische JavaScript-Programme aufgerufen werden (WENZ 2007, 317 ff.).

Innerhalb von Programmen besteht meist die Notwendigkeit Werte abzuspeichern, um diese an anderer Stelle oder zu einem späteren Zeitpunkt wieder aufrufen zu können. Diese Werte müssen dabei nicht unbedingt Zahlen sein, es kann sich dabei auch um Wörter oder andere Zeichenketten handeln. Das Speichern von Werten erfolgt mithilfe von Variablen. Diese müssen deklariert, d.h. dem System unter einen bestimmten Namen als Variablen bekannt gemacht werden. Deklarierten Variablen kann dann ein Wert zugewiesen werden. Wichtig ist dabei der Gültigkeitsbereich einer Variablen, man spricht auch von Lebensdauer. So genannte lokale Variablen sind nur innerhalb der Funktion, einem Teilbereich des Programms, gültig, in der sie deklariert wurden. Sie werden gelöscht, sobald die Funktion verlassen wird. Globale Variablen hingegen sind im gesamten Dokument gültig und können von jeder Funktion des Programms und zu jedem Zeitpunkt abgerufen werden (W3SCHOOLS 2007b).

JavaScript-Programme sind im Grunde eine Ansammlung von Anweisungen. Um den Programmablauf zu steuern, muss klar sein, wann welche Anweisung ausgeführt werden soll. Ein möglicher Zeitpunkt für die Ausführung einer Anweisung ist wie oben beschrieben ein durch den Benutzer ausgelöstes Ereignis, sofern dieses mit dem Aufruf einer Anweisung verknüpft ist. Ein solches Ereignis ruft häufig nicht nur eine einzige Anweisung auf, sondern löst in Abhängigkeit von dynamisch gespeicherten Werten ganze Ereigniskaskaden aus. Da dabei einzelne Anweisungen auch mehrfach aufgerufen werden können, sind Steuerelemente notwendig, die den Programmablauf koordinieren. Hierfür stehen in JavaScript bestimmte Kontrollstrukturen zur Verfügung. Sollen beispielsweise bestimmte Anweisungen immer wieder ausgeführt werden, solange bestimmte Bedingungen, etwa in Abhängigkeit von Wertebereichen, erfüllt sind, so werden so genannte Schleifen als Kontrollstrukturen eingesetzt. Eine andere Möglichkeit den Programmablauf zu kontrollieren, ist der Einsatz so genannter bedingter Abfragen. Hier wird in den Programmfluss eine Fallunterscheidung eingebaut, die für definierte Bedingungen bestimmte Anweisungen aufruft (GOODMAN 2006, 1139 ff.).

JavaScript ist wie bereits angesprochen eine objektorientierte Programmiersprache. Objektorientierung ist ein weiteres Paradigma der Softwareentwicklung. Grundlegende Ziele sind dabei, Programme flexibler zu gestalten sowie die Wiederverwendbarkeit einzelner Programmteile zu ermöglichen. Eine umfassende Einführung in die objektorientierte Softwarenetwicklung geben MEYER (1992) bzw. BRÜGGE & DUTOIT (2004).
Unter Objekten versteht man in der objektorientierten Programmierung bestimmte Bestandteile eines Programms. Objekte verfügen über definierte Eigenschaften und Methoden. Eigenschaften sind Attribute, die von außen abgefragt werden können. Methoden sind Aktionen oder Funktionen, die vom Objekt ausgeführt werden können (LAHRES & RAYMAN 2006, 63 ff.).
Ein Beispiel für ein Objekt, welches JavaScript standardmäßig bereit stellt, ist das Objekt Math. Dieses stellt mathematische Konstanten und Funktionen bereit. So können als Eigenschaften des Math-Objekts Werte wie pi oder natürliche Logarithmen abgerufen werden. Als Methoden stehen Rundungs-, Sinus-, Exponentialfunktionen und dergleichen zur Verfügung. Neben Objekten, die bereits als Komponenten in JavaScript enthalten sind, ist es möglich, eigene Objekte mit spezifischen Eigenschaften und Methoden zu definieren. JavaScript kann auch auf Objekte bzw. Elemente in HTML-Dokumenten zugreifen und diese verändern. Hierzu ist die Verwendung einer entsprechenden Schnittstelle notwendig. Diese stellt das Document Object Model (DOM) bereit, welches im nachfolgenden Kapitel erläutert wird.

6.1.5 Document Object Model - DOM

Das Document Object Model ist keine Programmiersprache, sondern eine Programmierschnittstelle. Solche Schnittstellen werden in der Fachsprache auch als API (Aplication Programming Interface) bezeichnet. Das Document Object Model ermöglicht Programmen den lesenden und schreibenden Zugriff auf HTML-Dokumente. Deren Inhalt, Struktur und Darstellung können auf diese Weise dynamisch verändert werden. Das DOM arbeitet plattformunabhängig und ist unabhängig von der genutzten Programmiersprache (LE HORS ET AL. 2004). Es bildet damit einen wesentlichen Baustein bei der Entwicklung von interaktiven Webanwendungen.

Spricht man vom Document Object Model, so meint man üblicherweise das vom World Wide Web Consortium standardisierte DOM. Dieses stellt eine Obermenge der traditionell von verschiedenen Web-Browsern verwendeten Modelle dar und ist in aktuellen Browsern zu großen Teilen implementiert (LUBKOWITZ 2005, 428 ff.)

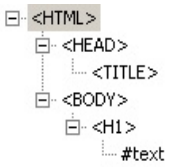

Abb. 24: DOM-Hierarchie eines einfachen HTML-Dokuments (eigene Darstellung)

HTML-Dokumente sind wie in Kapitel 6.1.2 beschrieben hierarchisch aufgebaut. Die hierarchische Struktur der einzelnen Elemente wird im DOM als Baumstruktur repräsentiert. Abbildung 24 zeigt einen solchen DOM-Baum anhand eines sehr einfach aufgebauten HTML-Dokuments, welches nur einen einzeiligen Text enthält. Die einzelnen Elemente werden in einer solchen Baumstruktur auch Knoten genannt. Somit beinhaltet jeder Knoten ein bestimmtes Element, die entsprechenden Attribute und den eigentlichen Inhalt. Das DOM stellt verschiedene Interfaces bereit, die ihrerseits über Eigenschaften und Methoden das Durchlaufen des DOM-Baums und den manipulativen Zugriff auf dessen einzelne Elemente regeln (GAMPERL 2007, 44 ff.).

6.1.6 Virtual Reality Modeling Language - VRML

VRML ist keine Programmiersprache wie JavaScript, sondern eine Beschreibungssprache für dreidimensionale Objekte. VRML kann als Pendant zu HTML für die Beschreibung von dreidimensionalen Inhalten im Internet verstanden werden.

Der Bedarf, auch die dritte Dimension über das Web transportieren zu können, entstand bereits relativ früh. Etwa um 1990 wurden erste Bemühungen unternommen, um ein 3D-Benutzerinterface für das Internet zu entwickeln. Daraus entstand in Zusammenarbeit mit der Firma Silicon Graphics schließlich eine erste Spezifikation für VRML, die jedoch nur die Darstellung von statischen 3D-Modellen vorsah. Da es bis zu diesem Zeitpunkt weder ernst zu nehmende proprietäre Entwicklungen noch einen offenen internationalen Standard gab, um 3D-Inhalte im Netz verfügbar zu machen, wurde die Weiterentwicklung von VRML durch ein unabhängiges, nicht gewinnorien-

tiertes Gremium voran getrieben. Die Arbeit dieses VRML-Consortiums, welches später im heutigen Web3D-Consortium aufgegangen ist, mündete schließlich in der Verabschiedung einer weiterentwickelten Version von VRML (WALSH & BOURGES-SÉVENIER 2001, 113 ff.). Diese wurde 1997 von der ISO unter dem Namen VRML 97 standardisiert (ISO 1997b).

Im Jahr 2002 wurde VRML um einige Komponenten erweitert. Mit dem VRML-EAI (External Authoring Interface) steht eine ISO-normierte Schnittstelle zur Verfügung, um VRML-Szenen mittels Java über externe Komponenten zu steuern (ISO 2002a). Um die Abbildung geographischer Inhalte zu erleichtern, wurde die optionale Komponente GeoVRML als Ergänzung zum bestehenden Spezifikationsumfang eingeführt (ISO 2002b). GeoVRML benötigt für den korrekten Betrieb bestimmte Java-Klassen. Wie noch erläutert werden wird, sind zur Darstellung von VRML-Dokumenten im Web-Browser zusätzliche Komponenten notwendig. Da Teile dieser Komponenten Java nur eingeschränkt unterstützt und die Verwendung von Java in Web-Browsern wie in Kapitel 6.1.4 beschrieben problematisch sein kann, spielen die Kombination von Java und VRML-EAI sowie GeoVRML in der breiten Anwendung nur eine untergeordnete Rolle.

Seit 2005 steht mit X3D (eXtensible 3D) ein von der ISO normierter Nachfolger für VRML 97 zur Verfügung (ISO 2005a). Die Hauptmerkmale von X3D sind der im Gegensatz zur monolithischen Struktur von VRML modulare Aufbau sowie die Basierung auf der Metasprache XML (Extensible Markup Language), die wiederum eine vereinfachte Teilmenge der bereits erwähnten Metasprache SGML ist (FUH & LI 2005, 573 bzw. GRAU 2001, 110). Neben der Notation in XML erlaubt X3D jedoch auch eine Codierung der Inhalte in der klassischen VRML-Syntax (ISO 2005b).

Bislang konnte sich X3D nicht auf breiter Ebene etablieren. Große Standard-Softwareanwendungen aus dem GIS- bzw. 3D-Bereich wie ESRI ArcGIS und Autodesk 3ds max, die häufig als Werkzeuge zur Generierung dreidimensionaler Geo-Daten eingesetzt werden (FREIWALD ET AL. 2006), bieten in der aktuellen Programmversion standardmäßig keinen X3D Export an. Der Export von Daten nach VRML gehört bei diesen Werkzeugen jedoch seit Jahren zum Standardumfang (LAMPRECHT 2002, 229). Darüber hinaus sieht das Open Geospatial Consortium (OGC) in einem Standardisierungsentwurf zur dreidimensionalen Web-Visualisierung von Geodaten VRML als Standardausgabeformat vor (OGC 2005). Dies sind Gründe dafür, dass gerade Projekte aus dem wissenschaftlichen Umfeld bei der Visualisierung dreidimensionaler Geo-Daten weiterhin auf den VRML-Standard setzen. Beispiele hierfür sind

aktuelle und laufende Projekte wie ‚Mainz Mobil 3D' (FISCHER & ZIPF 2006), ‚CityServer3D' (HAIST & ETZ 2005, 16 ff.) oder das von der EU geförderte, noch bis 2008 laufende Forschungsprojekt ‚VEPS' (VEPS 2005). Letzteres hat die Konzeption eines virtuellen Planungssystems für Raum und Umwelt zum Ziel, in dem Planungspartizipation und 3D-Visualisierung ebenfalls eine Rolle spielen. Eine ausführliche und vergleichende Diskussion von VRML 97, GeoVRML, X3D und anderen vektorbasierten Geodatenformaten ist bei FREIWALD & JANY (2005) zu finden.

Neben statischen Einzelobjekten können mit VRML ganze virtuelle Welten beschrieben werden. Die Möglichkeiten reichen von der reinen Darstellung dreidimensionaler Geometrien bis hin zur Steuerung des Verhaltens von Objekten zur Laufzeit des Programms. Letzteres bietet sich für die Integrierung von interaktiven Elementen und Animationen an. Das Einbinden von Lichtquellen und dreidimensionalen Klangräumen ist ebenfalls möglich. Die mit VRML beschriebenen dreidimensionalen Inhalte lassen sich mit externen Programmelementen verknüpfen. Hierfür kann der Quellcode von VRML mit JavaScript-Elementen bzw. VRML-Script, einer modifizierten Teilmenge von JavaScript, erweitert werden (RIEDL ET AL. 2002, 270 ff.). Der VRML 97-Standard ist damit grundsätzlich für die verschiedensten Anwendungsmöglichkeiten im Bereich 3D-Visualisierung, Interaktion und Internet geeignet. Auf Grund dieser Vielseitigkeit ist die Spezifikation entsprechend umfangreich (DIEHL 2001, 27 ff.).

Die nachfolgenden Abschnitte schildern jeweils einige spezifische Eigenschaften des VRML-Standards, die an späterer Stelle aufgegriffen werden. Eine umfassende Einführung in VRML geben MATSUBA & ROEHL (1996) bzw. AMES ET AL. (1997).
Der Inhalt von VRML-Dokumenten wird als Unicode im UTF-8 Format codiert, d.h. der Quelltext kann ähnlich wie ASCII-basierte Dokumente in jedem Texteditor geöffnet und verändert werden. Da VRML-Dokumente aufgrund der dreidimensionalen Dateninhalte im Vergleich zu HTML-Dokumenten sehr groß werden können, kann eine Kompression der Daten mittels Gzip vorgenommen werden, was eine schnellere Übertragung der Daten über das Internet ermöglicht (WALSH & BOURGES-SÉVENIER 2001, 250 ff.).
Im Vergleich zu klassischen CAD- und 3D-Programmen ist das dem VRML-Standard zu Grunde liegende Koordinatensystem verdreht. VRML basiert auf einem rechtshändigen, kartesischen Koordinatensystem. Bei diesem Raumbezugssystem verläuft die positive x-Achse horizontal nach rechts, die positive y-Achse vertikal nach oben und

die positive z-Achse horizontal nach vorn (Abb. 25). Anders als in CAD-Anwendungen, in denen die x- und die y-Achse die Horizontalebene definieren und Höhenwerte auf der z-Achse abgetragen werden, dient in VRML die y-Achse zur Notation der Höhe (SCHLÜTER 1998, 51 ff.). Dieser Umstand muss berücksichtigt werden, wenn dreidimensionale Geoobjekte aus bzw. in GIS- oder 3D-Programmen als VRML-Daten ex- bzw. importiert werden.

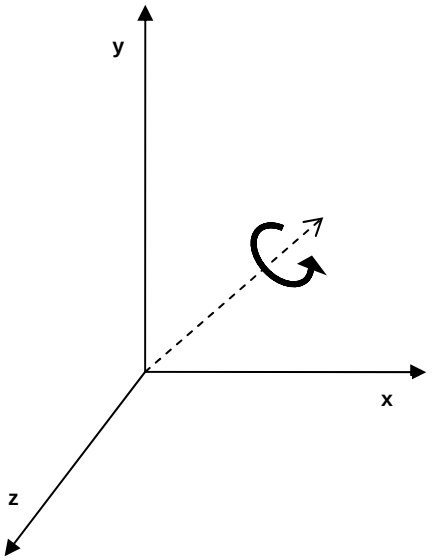

Abb. 25: Koordinatensystem und Rotationen im VRML-Standard (eigene Darstellung)

Um die Ausrichtung eines Objekts im dreidimensionalen Raum zu definieren nutzt VRML das Konzept des Achs-Winkels, auch Axis-Angle genannt (WALSH & BOURGES-SÉVENIER 2001, 628). Hierzu wird zunächst ein vom Ursprung des Koordinatensystems ausgehender Vektor gebildet; in Abbildung 25 als unterbrochener Pfeil dargestellt. Dieser stellt die Achse dar, um welche ein Objekt gedreht werden soll. Der Betrag, um den das Objekt gedreht werden soll, wird als Bogenmaß angegeben; in der Abbildung schematisch als kreisförmiger Pfeil abgebildet. Für eine Rotation in VRML sind somit vier Angaben notwendig: x-, y- und z-Wert mit einem Wertebereich von -1 bis +1 zur Definition der Achse sowie ein Winkel a mit einem Wertebereich von 0 bis 2 pi bzw. -1pi bis +1pi (LEA ET AL. 1996, 308 ff.). Im Gegensatz zur Rotationsdefinition mittels Eulerwinkeln, bei denen in einer bestimmten Reihenfolge um drei Achsen

gedreht wird, ist das in VRML verwendete Prinzip für den Entwickler wie auch für den Benutzer wenig intuitiv. Deshalb werden zur Ausgabe von Winkeln in der Online-Plattform entsprechende Transformationen durchgeführt (vgl. Kap. 6.5). Die mathematischen Grundlagen sowie eine Einführung in die verschiedenen Möglichkeiten der Definition von Rotationen im euklidischen Raum sind bei BAKER (2007) bzw. LEUPOLD (1990) zu finden.

VRML bildet die Beschreibung einer dreidimensionalen virtuellen Welt, auch Szene genannt, in einem so genannten Szenegraphen ab. Der Szenegraph ordnet alle für die Szene relevanten Elemente wie Geometrie und Aussehen einzelner Objekte sowie Interaktionseigenschaften und Animationen hierarchisch in einer Baumstruktur an. Die einzelnen Elemente dieser Anordnung werden als Knoten bezeichnet. Diese können ihrerseits weitere, hierarchisch nachgeordnete Unterknoten enthalten, so dass in der Baumstruktur einzelne Zweige gebildet werden (HASE 1997, 10 ff.). Diese als Baumstruktur abgebildete Hierarchie ist mit der Anordnung einzelner HTML-Elemente in einem DOM-Baum vergleichbar.
Es existiert eine Vielzahl unterschiedlicher Arten von Knoten. Grundsätzlich sind drei Arten von Knoten zu unterscheiden. Gruppenknoten dienen oft der Gruppierung von Elmenten und können eine beliebige Anzahl von Kindknoten enthalten. Kindknoten können entweder weitere Gruppenknoten oder so genannte Blattknoten sein. Blattknoten definieren die eigentlichen Elemente in einem VRML-Dokument. Blattknoten enthalten ihrerseits so genannte untergeordnete Knoten, die das Aussehen von sichtbaren Objekten beschreiben. Zur Beschreibung der Eigenschaften von Knoten sowie zur Speicherung von Parametern stehen in den Knoten jeweils so genannte Felder zur Verfügung.
Viele Knoten besitzen die Eigenschaft, Ereignisse, events genannt, zu senden oder zu empfangen. Mit Hilfe dieser events kommunizieren Knoten miteinander und tauschen so Werte aus oder benachrichtigen einander, wenn sich bestimmte Parameter eines Knotens verändert haben. Ereignisse, die ein Feld verändern können, werden dabei als eventIn, bezeichnet. Ein Beispiel dafür ist das Ereignis set_position, welches die Werte des Feldes position verändert. Ein Ereignis, das die Änderung eines Feldes bekannt gibt, wird eventOut genannt. Wurden die Werte des Feldes position verändert, so wird das Ereignis position_changed generiert (HASE 1997, 30 ff.). Die Kommunikationswege zwischen bestimmten Knoten werden durch so genannte Routen bekannt gemacht. Routen sind keine Knoten. Routen stellen die Verbindung zwischen bestimm-

ten Feldern von Knoten her und sorgen so für die Übertragung von Ereignissen zwischen diesen Feldern (DIEHL 2001, 37 ff.).
Vor dem Hintergrund der events bilden die so genannten Sensoren eine besondere Art von Knoten in der VRML-Spezifikation. Sie registrieren bestimmte benutzerspezifische Interaktionen und generieren daraus events, die ihrerseits weitere Ereignisse auslösen können. Um solche Ereigniskaskaden zu steuern bzw. auszuwerten, bietet sich das Einbinden spezifischer JavaScript-Programme an. Der Sensorknoten Proximity-Sensor erzeugt Ereignisse, wenn sich der Avatar in der 3D-Szene bewegt. Diese Ereignisse enthalten Angaben über die Position und Ausrichtung des Avatars im lokalen Raumbezugssystem des Sensorknotens. Der Sensorknoten Planesensor erfasst Ziehbewegungen der Maus und erzeugt Ereignisse, die die auf eine horizontale Ebene projizierte Position des Mauszeigers in der Szene liefern. So genannte TouchSensoren erfassen ebenfalls die Position des Mauszeigers in der 3D-Szene. Bei Berührung geometrischer Objekte können verschiedene Ereignistypen erzeugt werden (ROEHL ET AL. 1997, 23 ff.).
Eine weitere spezielle Art von Knoten sind die bindbaren Knoten. Hiervon existieren mehrere Typen. Zu einem Zeitpunkt kann in einer Szene nur ein bindbarer Knoten des jeweiligen Typs aktiv sein. Beispiele sind die Knoten NavigationInfo oder Background. Der Knoten NavigationInfo enthält verschiedene Informationen darüber, wie der Avatar beschaffen sein soll und wie er sich in der Szene bewegen soll. Der Knoten Background definiert den Hintergrund in einer 3D-Szene, sofern er nicht durch spezifische geometrische Objekte verdeckt ist. Hiermit lässt sich beispielsweise ein Himmel simulieren. Innerhalb der Klasse der bindbaren Knoten spielt der Knoten Viewpoint für die Navigation innerhalb der Szene eine besondere Rolle. Mittels eines Viewpoints wird die Position und Ausrichtung eines Blickpunktes in einer Szene definiert. Dies ist vergleichbar mit der Installation einer virtuellen Kamera. Neben der Position und Orientierung im lokalen Koordinatensystem kann das Sichtfeld eines Viewpoints bestimmt werden (HASE 1997, 222 ff.).
Für die Beschreibung geometrischer Objekte stehen verschiedene Knoten zur Verfügung. Geometrische Primitive wie Würfel, Kugeln oder Kegel können mit spezifischen Knoten und durch Definition charakteristischer Parameter wie Kantenlänge, Radius usw. erzeugt werden. Komplexere geometrische Objekte werden aus einzelnen Polygon-Dreiecken zusammengesetzt. Hierbei werden die Position einzelner Scheitelpunkte sowie die Verbindung zwischen diesen beschrieben. Auf diese Weise werden zusammenhängende Flächen definiert. Diese können mit Texturen belegt werden. Wie

diese Texturen auf den Flächen dargestellt werden sollen, wird in so genannten Material-Knoten notiert.
Die VRML-Spezifikation sieht das externe Referenzieren von Dateien vor. Der Inhalt einer 3D-Szene kann somit auf mehrere VRML-Dokumente verteilt werden (DICKMANN 2004, 102). Externe Dateien werden über so genannte Inline-Knoten referenziert. Diese Möglichkeit wird für die Implementierung der Online-Plattform genutzt, um geometrische von thematischen und verwaltungstechnischen Inhalten bzw. Informationen zu trennen.
Im Gegensatz zu HTML-Dokumenten können VRML-Dokumente von Web-Browsern nicht ohne zusätzliche Komponenten interpretiert werden. Welche Komponenten benötigt werden, wird im nachfolgenden Kapitel erläutert.

6.1.7 ActiveX

Web-Browser benötigen zur Darstellung der Inhalte von VRML-Dokumenten zusätzliche Komponenten. Diese werden bei den Browsern der Netscape/Mozilla-Familie Plugins bzw. Extensions genannt. Beim Microsoft Internet Explorer werden die zur Darstellung fremder Dokumentformate notwendigen Komponenten ActiveX Controls genannt. Als Entwicklungsplattform wurde wie in Kapitel 6.1.2 beschrieben der weltweit meistgenutzte Web-Browser, der Internet Explorer der Firma Microsoft, ausgewählt. Konzeption und Implementierung der Online-Plattform sind grundsätzlich aber nicht an einen bestimmten Browser gebunden. Das Online-Informationssystem ist sowohl auf Browsern der Netscape/Mozilla-Familie wie beispielsweise Firefox als auch auf anderen Browsern wie dem Internet Explorer lauffähig.

ActiveX ist eine Sammlung verschiedener Technologien, die auf dem Component Object Model (COM) von Microsoft basieren (MICROSOFT 2007b). Ziel dieses Models ist es, Programmcode in wieder verwendbare Bausteine zu zerlegen (ALTENSCHMIDT 1999). Bei der Verwendung der ActiveX-Technologie unterscheidet man die drei Komponenten Control, Automation und Documents (MICROSOFT 2007c). Die Komponente ActiveX Control ist ein Programm oder Modul, welches sich in HTML-Dokumente einbinden lässt. Das ActiveX Control kommuniziert dabei über die OLE-Schnittstelle (Object Linking and Embedding) mit anderen über den Web-Browser erreichbaren Komponenten (MASLO ET AL. 2005, 866 ff.).

Mittels ActiveX ist es möglich, die unterschiedlichsten Multimedia-Komponenten als Objekte direkt in Web-Browser einzubinden. Gegenüber der bei der Netscape/Mozilla-Browserfamilie eingesetzten Plugin-Technik gilt ActiveX als das intelligentere Modell (KERSKEN 2003, 864). Der Browser lädt die benötigten Komponenten automatisch herunter und installiert diese zur Laufzeit, ohne dass ein Neustart des Web-Browsers notwendig wird.
Zur Darstellung der VRML-Dokumente in der Online-Plattform wird das ActiveX Control Cortona des Herstellers ParallelGraphics verwendet (PARALLELGRAPHICS 2007a). Cortona ist eine der am häufigsten verwendeten Erweiterungen zur Darstellung von VRML-Dokumenten (DECKER & RUNDE 2003, 68 ff.). Eine im Vergleich zu anderen VRML-Controls deutlich geringere Dateigröße verkürzt die Ladezeiten. Zur Entwicklung von Software-Applikationen steht ein umfangreiches Software Development Kit zur Verfügung (PARALLELGRAPHICS 2007b). Für den nichtkommerziellen Einsatz ist Cortona frei verfügbar und kann in alle gängigen Web-Browser problemlos eingebunden werden. Cortona findet gerade in wissenschaftlichen Projekten eine häufige Anwendung (FISCHER & ZIPF 2006). Eine vergleichende Gegenüberstellung aktueller VRML-Controls bzw. Extensions geben GRIGORIEVA (2005) bzw. MUCHMORE (2006).

6.2 Systemarchitektur

Dem System liegt eine Client-Server Architektur zu Grunde (vgl. Kap. 6.1.1). Dem Server kommt dabei in erster Linie die Aufgabe zu, die für das System notwendigen Datenressourcen zur Verfügung zu stellen. Dies sind sämtliche für dieses Projekt vom Verfasser erstellten 3D-Daten, thematische Basisinformationen zu einzelnen 3D-Objekten sowie die zum Betrieb des Informationssystems notwendigen HTML- und JavaScript-Dokumente.
Da die Interaktionen des Benutzers mit dem System gemäß den Anforderungen in Kapitel 4.2.3 möglichst ohne spürbare Verzögerung ablaufen sollen, wurde bereits bei der Auswahl der Programmiersprache und Dateiformate darauf geachtet, dass ein hohes Maß an clientseitiger Datenverarbeitung ermöglicht wird. Dies setzt bei dreidimensionalen Daten grundsätzlich rechenstarke Hardware des Client-Rechners voraus. Aufgrund des speziell für unterschiedliche Anforderungen entwickelten 3D-Modells, konnten diese rechenintensiven 3D-Daten vollautomatisch für die Anwendung auf gering performanter Hardware optimiert werden. Daher sind mehrere Jahre alte Standard-Rechner ausreichend schnell, um das Informationssystem aufzurufen.

Da die dreidimensionale Echtzeit-Visualisierung und der Großteil der in Kapitel 6.3 beschriebenen Funktionalitäten des Systems auf Clientseite ablaufen, wird zudem der Anspruch erfüllt, das System auch unabhängig von der Verfügbarkeit des Internets bzw. des Servers betreiben zu können. Die Datengrundlage muss zwar in diesem Fall auf den lokalen Rechner kopiert werden, es ist aber nicht notwendig, aufwendig einen lokalen Server aufzusetzen wie dies beispielsweise beim Offline-Betrieb von Standard Web-GIS-Anwendungen notwendig ist. Die Abfrage von Objektinformationen aus Datenbanken oder ein Zugriff auf mit dem System verbundene Online-Ressourcen wie beispielsweise Webseiten, Partizipationsforen oder Baupläne sind bei der lokalen Installation des Informationssystems nicht möglich.

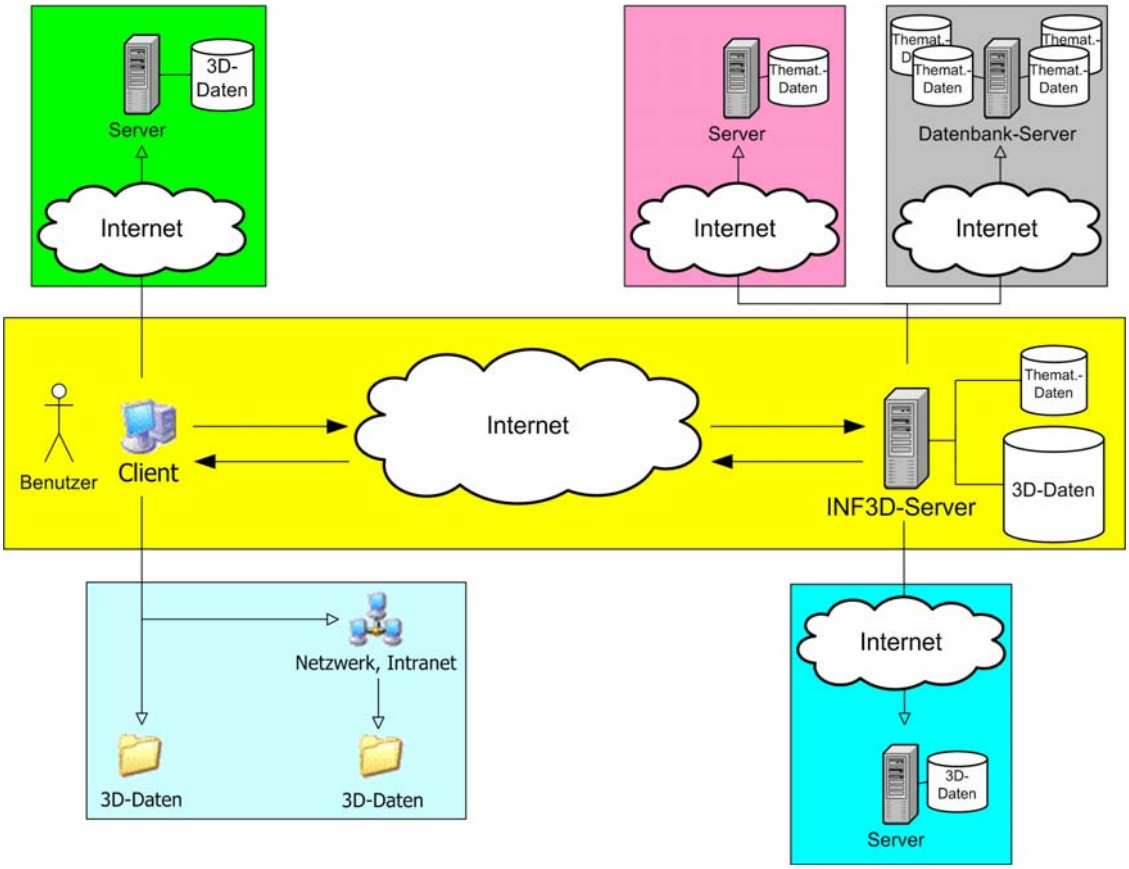

Abb. 26: Systemarchitektur (eigener Entwurf und Darstellung)

Abbildung 26 zeigt eine schematische Darstellung der Systemarchitektur. Der gelbe Bereich in der Abbildung stellt deren Kern dar. Ein Benutzer ruft mithilfe eines Client-Rechners das Informationssystem auf. Der Client-Rechner greift dabei über das Inter-

net auf einen speziell für das System eingerichteten Server (INF3D-Server) zu, der die zur Initialisierung des Systems notwendigen Dokumente an den Client überträgt. Dies sind zum einen diejenigen Dokumente, die für die Darstellung und Funktionalität der Benutzeroberfläche notwendig sind. Diese Dokumente werden durch den Internetbrowser des Clients interpretiert und dargestellt. Zum anderen sind dies Dokumente, die die dreidimensionalen Daten sowie die damit verknüpften thematischen Informationen enthalten.

Neben thematischen Basisinformationen, die direkt auf dem INF3D-Server vorgehalten werden, greift das System auf externe thematische Daten zu, die auf anderen Servern im Internet liegen (Abb. 26, rosafarbener Bereich). Hierfür sind in den thematischen HTML-Dokumenten auf dem INF3D-Server entsprechende Hyperreferenzen enthalten. Hyperreferenzen stellen über Verweise (Hyperlinks) Beziehungen zu anderen Quellen her. Über solche Hyperlinks sind beispielsweise Webseiten von Instituten und Kliniken oder PDF-Dokumente des Heidelberger Gemeinderats mit planungsbezogenen Inhalten erreichbar.

Das Konzept sieht ebenfalls die Verbindung zu Datenbanken mit spezifischen Informationen vor (Abb. 26, grauer Bereich). Dieser Teil ist im Prototyp nicht realisiert, da entsprechende Datenbanken zwar zur Verfügung stehen, der Zugriff darauf aber auf einen bestimmten Benutzerkreis eingeschränkt ist. Die technische Möglichkeit, solche Datenbanken anzuschließen bzw. auf deren Inhalte zuzugreifen wird in Kapitel 6.5.7 aufgezeigt.

Die Dokumentstruktur der 3D-Daten wurde so gewählt, dass die Daten auch ausgelagert werden können (vgl. Kap. 6.4). Dies ist von Vorteil, wenn es um Daten geht, die sich regelmäßig ändern; wie z.B. 3D-Modelle verschiedener Planungsvarianten. Damit haben Dritte die Möglichkeit, die jeweils aktuelle Version der Daten auf einem eigenen Server im Internet bereit zu stellen. Das INF3D-Informationssystem holt diese vollautomatisch dort ab und fügt sie mit den übrigen 3D-Daten zusammen (Abb. 26, türkisfarbener Bereich). Vorraussetzung ist, dass die Daten in dem vom Informationssystem verwendeten Dateiformat VRML sowie entsprechend georeferenziert zur Verfügung stehen.

Das Informationssystem kann um benutzerdefinierte dreidimensionale Inhalte erweitert werden, die sich nicht auf dem INF3D-Server befinden. Der Benutzer hat die Möglichkeit lokal auf seinem System oder im daran angeschlossenen Netzwerk abgespeicherte 3D-Daten in das Informationssystem hinein zu laden (Abb. 26, hellblauer Bereich). Dies bietet sich zum Beispiel für Planungsämter oder Architekturbüros an, die

ihre eigenen Daten mit denen des Informationssystems kombinieren möchten. Alternativ besteht die Möglichkeit, auf im Internet zur Verfügung stehende Daten zuzugreifen und diese in das Informationssystem zu laden (Abb. 26, grüner Bereich). Ein mögliches Szenario ist ein Benutzer, der sich die von verschiedenen Planungsbüros oder Ämtern zur Verfügung gestellten Varianten eines Vorhabens im Informationssystem darstellen lassen möchte. Vorraussetzung ist in beiden Fällen, dass die Daten als entsprechend georeferenziertes VRML-File vorliegen. Die auf einem dieser beiden Wege in das Informationssystem integrierten Daten sind nur für den Benutzer auf Clientseite sichtbar. Die eigentlichen bzw. vorgegebenen Inhalte des Systems auf Serverseite werden dadurch nicht verändert.

Die Systemarchitektur bietet somit drei verschiedene Wege, 3D-Daten oder thematische Inhalte zu integrieren. Der direkte Weg ist die zentrale Datenspeicherung auf dem INF3D-Server. Alternativ können vom Benutzer lokale Daten eingebunden werden. Die dritte Möglichkeit ist die Verknüpfung mit Daten aus dem Internet. Letzteres geschieht über die Anbindung von Datenbanken oder über Hyperreferenzen. Alle drei Arten sind beliebig kombinierbar. Die Systemarchitektur ist darüber hinaus so gestaltet, dass das Informationssystem nicht nur auf direktem Weg über die Eingabe der Internetadresse aufgerufen, sondern auch selbst als Hyperreferenz von anderen Internetseiten aus angesteuert werden kann. So ist es möglich, in eine beliebige Webseite einen Hyperlink einzubauen, der das Informationssystem vollautomatisch mit definierbaren Einstellungen aufruft. Eine Anwendungsmöglichkeit hierfür sind Hyperlinks auf den Webseiten universitärer Einrichtungen, die das Informationssystem unter Aufruf einer bestimmten Kameraposition starten. Dies wäre in der Webseitenrubrik „So finden Sie uns“ eine Alternative zu klassischen Karten, die Auskunft über die räumliche Lage und Erreichbarkeit eines Gebäudes geben. Vor dem Hintergrund der baulichen Planung und Öffentlichkeitsbeteiligung sind entsprechende Hyperlinks auf den Webseiten der Planungsbehörden vorstellbar, die das Informationssystem mit definierten Einstellungen starten. Dem Benutzer können so gezielt und vollautomatisiert bestimmte Bereiche und Inhalte des 3D-Modells präsentiert werden.

6.3 Graphische Benutzeroberfläche und Funktionalitäten

Die graphische Benutzeroberfläche stellt den Teil des Informationssystems dar, welcher für den Benutzer permanent sichtbar ist. Neben denjenigen Elementen, die der

rezipierenden Informationsvermittlung dienen, enthält die Oberfläche Schnittstellen, die dem Nutzer die Interaktion mit dem System und dessen Inhalten erlauben.
Es ist nicht Ziel dieser Arbeit, neue Erkenntnisse über die Gestaltung von Schnittstellen zwischen Mensch und Maschine zu sammeln oder ein Best-Practice Szenario für optimale Gestaltung solcher Schnittstellen aufzustellen. Das Thema Usability, d.h. die Gebrauchstauglichkeit von Online-Informationssystemen wird umfassend bei NIELSEN ET LORANGER (2006) behandelt. Mit der Ergonomie von Benutzerschnittstellen insbesondere von webbasierten Geo-Informationssystemen beschäftigt sich TONNIER (2002). Auf die Interaktion speziell mit dreidimensionalen Daten geht PREIM (1999) ein.

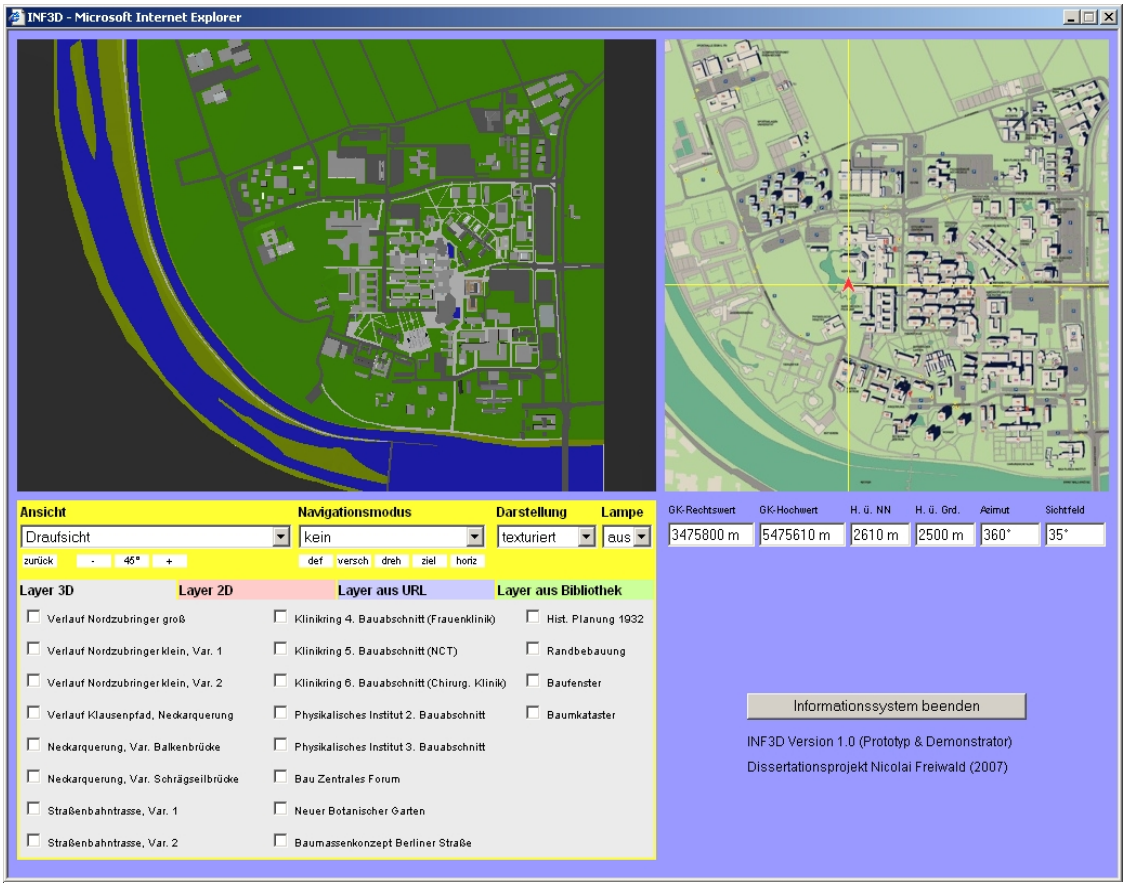

Abb. 27: Graphische Benutzeroberfläche (Screenshot, eigene Darstellung)

Wie bereits erwähnt steht die realisierte Ausbaustufe des Systems auf dem Niveau eines Protoypen. Dessen volle Funktionsfähigkeit und Online-Tauglichkeit wurden bereits erfolgreich getestet (vgl. CD im Anhang). Dieser Prototyp vereint sämtliche unter

Kapitel 6.5 beschriebenen Funktionsmodule in einer einzigen Benutzerschnittstelle. Dies mag sich nachteilig auf die Übersichtlichkeit der graphischen Benutzeroberfläche auswirken, wird aber für den Prototyp zu Gunsten der vollen Demonstrierbarkeit sämtlicher Module in Kauf genommen. Dennoch wurden Grundregeln hinsichtlich sinnvoller Gestaltung von graphischen Benutzeroberflächen und intuitiver Bedienbarkeit beachtet. Die Grundlagen, nach denen einzelne Bedienelemente gestaltet und innerhalb der graphischen Benutzeroberfläche angeordnet wurden, finden sich in GALITZ (2002) bzw. in MICROSOFT PRESS (1995). Nachfolgend werden die Einteilung der Benutzeroberfläche sowie deren einzelne Interaktionswerkzeuge beschrieben. Einige dieser Werkzeuge lassen sich mit den Medien Text und Bild nicht zufriedenstellend beschreiben. Deshalb liegt der Arbeit im Anhang eine CD bei, die sämtliche Funktionalitäten des Systems in einem Film zeigt.

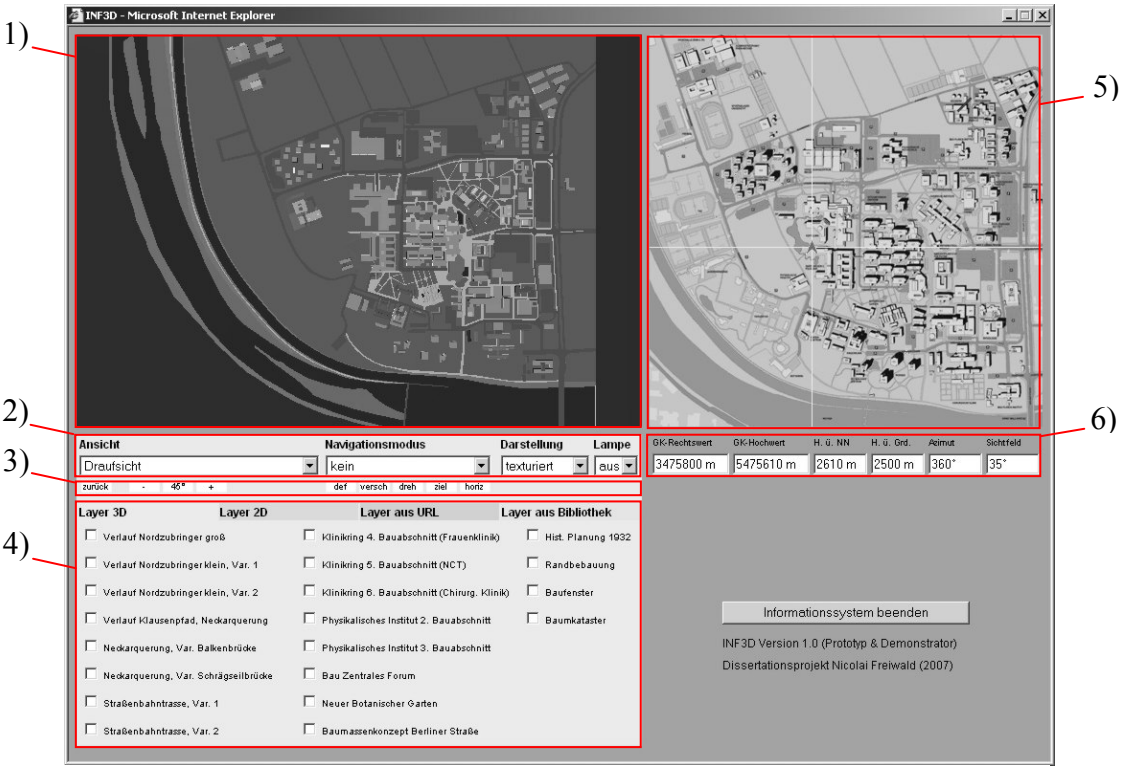

Abb. 28: Teilbereiche der graphischen Benutzeroberfläche (eigene Darstellung)

Abbildung 27 stellt die Benutzeroberfläche unmittelbar nach der Initialisierung des Systems dar. Grob lassen sich drei Bereiche unterscheiden. Der Bereich in der linken oberen Hälfte des Fensters gibt die dreidimensionalen Inhalte wieder. Im Bereich dar-

unter sind die Elemente zur Interaktion mit der Szene angeordnet. Die rechte Hälfte der Benutzeroberfläche wird dominiert von Elementen, die dem Benutzer die aktuelle Position zeigen. In Abbildung 28 ist die Feingliederung der Oberfläche in folgende sechs Teilbereiche dargestellt:

1. 3D-Fenster
2. Werkzeuge zur elementaren Navigation und Interaktion
3. Werkzeuge zur erweiterten Steuerung von Ansicht und Navigation
4. Steuerung der Szeneninhalte
5. Positionsanzeige und absolute Positionierung des Avatars
6. Ausgabe der Koordinaten und Blickfeldeigenschaften

Diese Teilbereiche werden in den nachfolgenden Unterkapiteln detailliert beschrieben. Die Teilbereiche 5 und 6 sind aufgrund inhaltlicher Überschneidungen zu einem Kapitel zusammengefasst.

6.3.1 3D-Fenster

In Bereich 1 der Abbildung 28 wird das digitale Computermodell angezeigt. Man spricht in diesem Zusammenhang auch von 3D-Szene. Diese Szene ändert sich in Abhängigkeit von Benutzereingaben bzw. -interaktionen. Objekte in der Szene können mit dem Mauszeiger berührt und angeklickt werden. Bei Berührung werden zunächst Kurzinformationen zum entsprechenden Objekt direkt im 3D-Fenster eingeblendet (Abb. 29). Klickt der Benutzer ein Objekt an, so wird ein zusätzliches Browserfenster geöffnet, welches weitergehende Informationen zum Objekt bereit stellt (Abb. 30). Diese Informationen können entweder direkt auf dem INF3D-Server vorgehalten werden oder über Datenbankabfragen von externen Servern eingebunden werden. Beide Möglichkeiten sieht die Systemarchitektur vor (vgl. Kap. 6.2). Der Umstand, dass diese Funktion hier vergleichsweise knapp beschrieben ist, darf nicht darüber hinwegtäuschen, dass es sich dabei neben der 3D-Visualisierung um eine der Kernfunktionen des Systems handelt; nämlich die Verknüpfung von 3D-Objekten mit beliebigen thematischen Informationen. Das Konzept sieht vor, dass Informationen dynamisch aus Datenbanken abgefragt werden. Die dreidimensionalen Geometrien sind mit eindeutigen IDs ausgestattet, um so eine Informationszuordnung sicher zu stellen. Über das zusätzliche Browserfenster kann außerdem die Verbindung mit einem Facility-Management-System oder objektspezifischen Threads in einem Online-Forum für Planungspartizi-

pation hergestellt werden. Die Implementierung ist in den Kapiteln 6.4 bzw. 6.5.7 beschrieben.

Abb. 29: Einblenden von Kurzinformationen zu Objekten durch Berührung mit dem Cursor (Screenshot aus dem Informationssystem, eigene Darstellung)

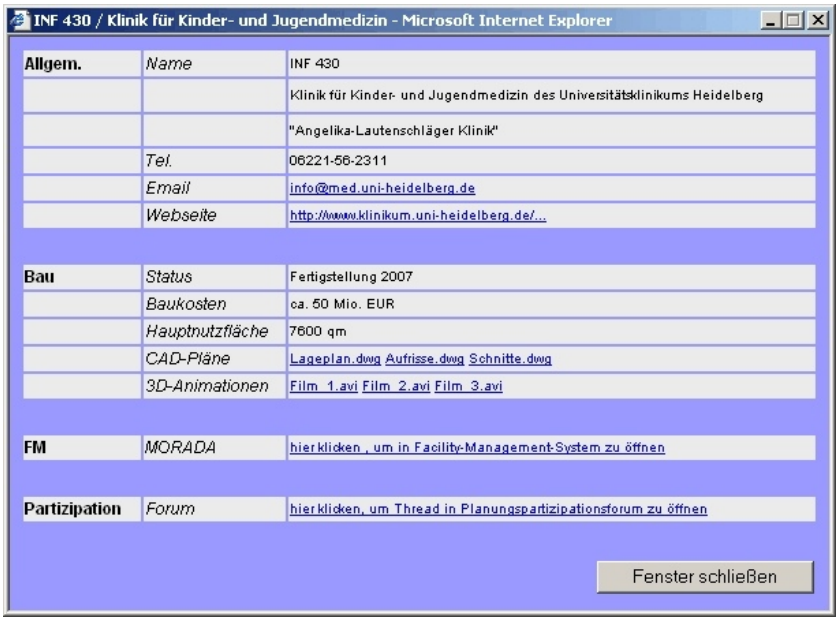

Abb. 30: Zusätzliches Browserfenster mit Informationen, Links zu Dateien und Webseiten sowie objektspezifischen Verknüpfungen zu einem möglichen Gebäudeverwaltungssystem und Forum für Planungspartizipation (Screenshot aus dem Informationssystem, eigene Darstellung)

6.3.2 Elementare Navigation und Interaktion

Der Bereich 2 der Benutzeroberfläche in Abbildung 28 stellt über so genannte Auswahllisten verschiedene Werkzeuge zur Verfügung, die es dem Nutzer auf sehr einfache Art erlauben in der Szene zu navigieren bzw. die Darstellung der Szeneninhalte an seine Bedürfnisse anzupassen. Abbildung 31 zeigt in einer Bildmontage die geöffneten Auswahllisten. Der Gebrauch solcher Auswahllisten ist intuitiv, zumal dem Benutzer diese Art der Auswahl von Optionen aus sämtlichen windowsbasierten Computer-Anwendungen bekannt sein dürfte.

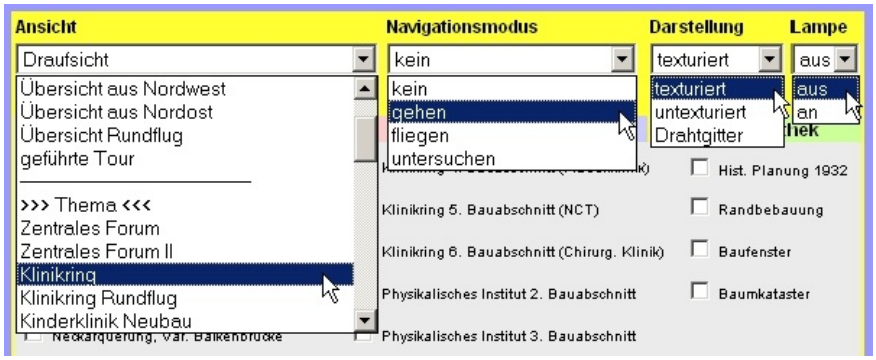

Abb. 31: Optionen in den Auswahllisten der Basiswerkzeuge (bearbeiteter Screenshot aus dem Informationssystem, eigene Darstellung)

Die Auswahlliste „Ansicht“ erlaubt die Ansteuerung in der Szene vorgegebener virtueller Kameras, die bestimmte Ausschnitte des dreidimensionalen Modells zeigen. Diese Kameras werden auch Viewpoints genannt. Die Auswahl erfolgt auf Grundlage beschreibender Texte. Diese sind in die Kategorien „Übersicht“, „Thema“ und Gebäude“ gegliedert. Während unter der ersten Kategorie diejenigen virtuellen Kameras zusammengefasst sind, die das gesamte Modell aus entsprechender Entfernung zeigen, sind in der Kategorie „Gebäude“ diejenigen aufgeführt, welche einzelne Gebäude zeigen. Die Kategorie „Thema“ führt eine Reihe von Einträgen auf, hinter denen sich bestimmte Teile des Modells verbergen, die im Fokus aktueller oder langfristiger Planungen stehen. Als Beispiel sind hier der Heidelberger Klinikring, die Erschließung des zentralen Forums, der Ausbau der Berliner Straße oder die Varianten der fünften Neckarquerung zu nennen. Neben statischen Viewpoints enthält die Auswahlliste auch animierte Kamerafahrten. Diese zeigen ein entsprechendes Gebäude oder ein Thema in animierter Form. Die Besonderheit dabei ist jedoch, dass der Benutzer trotzdem voll mit der Szene interagieren, d.h. beispielsweise Abfragen über Gebäude durchführen

oder den Blickwinkel anpassen kann. Dies ist bei herkömmlichen Computeranimationen oder Filmen von virtuellen 3D-Modellen nicht möglich. Über den Eintrag „geführte Tour“ in der Kategorie „Übersicht“ wird eine vollautomatische Führung zu allen in der Szene enthaltenen Viewpoints gestartet. Unabhängig davon, wo sich der virtuelle Betrachter in der Szene gerade befindet, startet die Führung immer mit derjenigen Ansicht, welche auch bei der Initialisierung des Systems voreingestellt ist. Einer vorbestimmten Reihenfolge gemäß wird dann alle fünf Sekunden der nächste Viewpoint aufgerufen. Der Wechsel zwischen zwei Ansichten erfolgt animiert, so dass der Betrachter die Orientierung behält. Sind nach etwa vier Minuten alle vordefinierten Viewpoints aufgerufen, beginnt die Führung von vorne. Der Benutzer kann durch die Auswahl eines anderen Eintrags aus der Liste „Ansicht“ die Tour zu jedem Zeitpunkt abbrechen. Das System bleibt zudem während dieser Führung voll interaktionsfähig; Objektinformationen können abgefragt, zusätzliche Inhalte eingeblendet oder die Blickrichtung verändert werden.

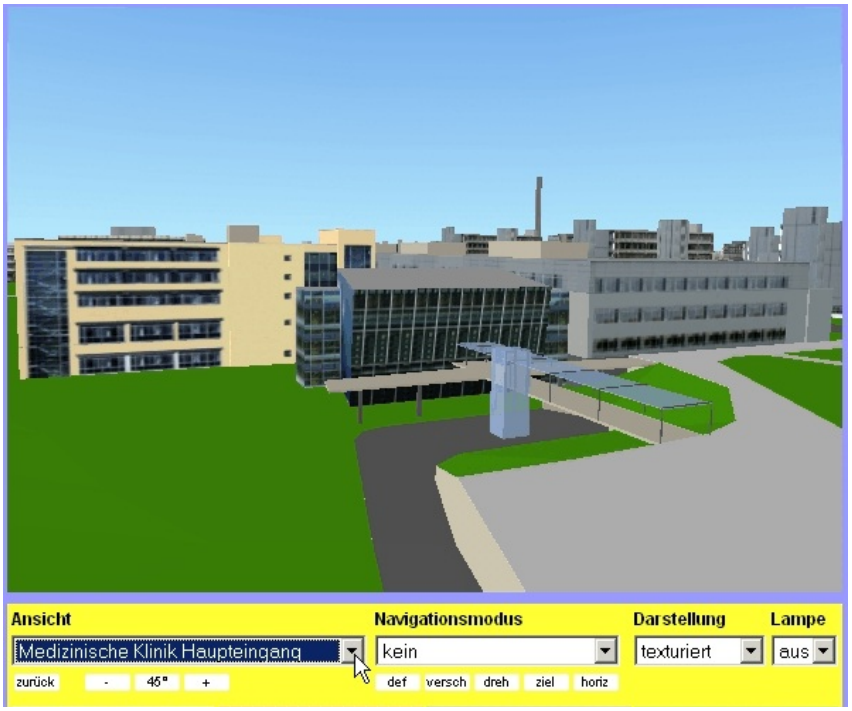

Abb. 32: Auswahl eines Viewpoints und entsprechende Ansicht in der 3D-Szene (Screenshot aus dem Informationssystem, eigene Darstellung)

Neben den Kategorien, die diese Auswahlliste zur Verfügung stellt, sind weitere wie zum Beispiel Gebäudenummern nach dem Muster INF „123“ denkbar. Sämtliche Ein-

träge dieser Liste können dem Nutzer auch in Form einer Such-Funktion zur Verfügung gestellt werden, bei der er einen Freitext eingeben kann und eine entsprechende Funktion den passenden Viewpoint aktiviert. Abbildung 32 zeigt am Beispiel der Medizinischen Klinik die Auswahl eines Viewpoints und die dazugehörige Ansicht in der 3D-Szene.
Die Auswahlliste „Navigationsmodus“ stellt dem Benutzer verschiedene Möglichkeiten zur Verfügung, wie er die Position des Avatars in der Szene verändern kann. Dies geschieht ausgehend von der aktuellen Ansicht. Es stehen die Modi „gehen“, „fliegen“ und „untersuchen“ zur Wahl (Abb. 31). Ist ein Modus aktiv, so kann durch Ziehbewegungen mit der Maus im 3D-Fenster der Avatar in der Szene bewegt werden. Die Modi „gehen“ und „fliegen“ sind selbsterklärend und intuitiv. Beim Modus „untersuchen“ wird der Avatar auf einer gedachten Kugeloberfläche bewegt, deren Mittelpunkt sich im Zentrum der gesamten 3D-Szene befindet.

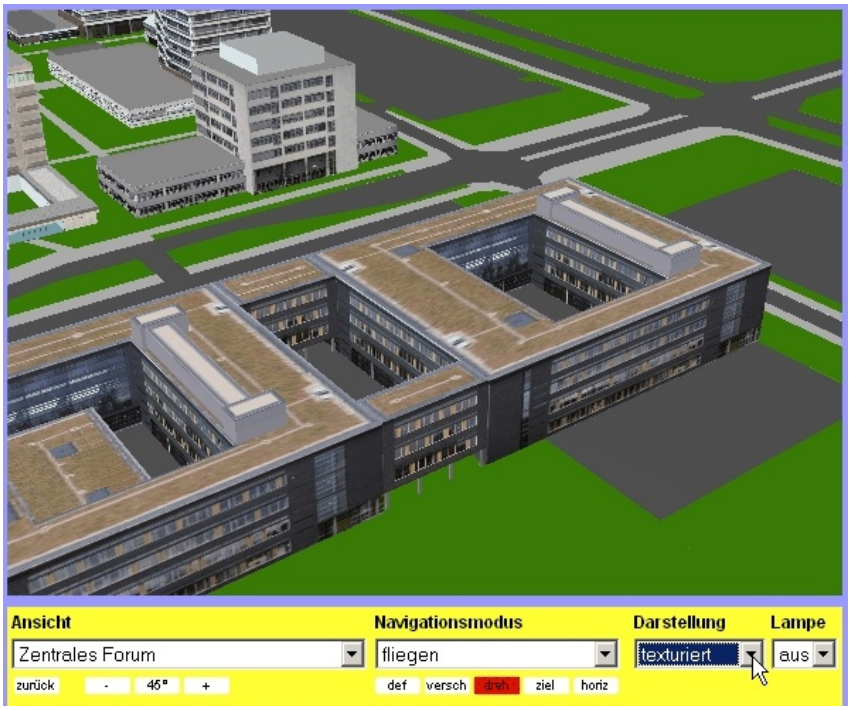

Abb. 33: Texturierte Darstellung der Szeneninhalte (Screenshot aus dem Informationssystem, eigene Darstellung)

Über die Optionen der Auswahlliste „Darstellung“ wird die Darstellung der Szene verändert. Als Standard ist die Art ausgewählt, bei der die Objekte der Szene „texturiert“ sind. Die Objekte werden dabei mit den ihnen zugeordneten Bildinformationen wie

Fassadenfotos angezeigt (Abb. 33). Die Option „untexturiert“ zeigt die Objekte ohne diese Bildinformationen an (Abb. 34). Dies Darstellungsart kann beispielsweise für schematische Ansichten sinnvoll sein. Auf sehr rechenschwacher Hardware kann diese Option gewählt werden, um eine flüssige Darstellung der Szeneninhalte zu erreichen. Der Eintrag „Drahtgitter“ stellt die äußeren Begrenzungskanten der in der Szene enthaltenen Geometrie dar.

Abb. 34: Nicht texturierte Darstellung der Szeneninhalte (Screenshot aus dem Informationssystem, eigene Darstellung)

Die Auswahlliste „Lampe“ bietet die Möglichkeit eine an den Avatar gebundene Lichtquelle zu aktivieren. Zwar ist die 3D-Szene gleichmäßig ausgeleuchtet, dennoch kann es Situationen geben, in denen die Inhalte im 3D-Fenster zu dunkel erscheinen. Dies ist zum Beispiel dann der Fall, wenn Innenräume betreten werden. Abbildung 35 zeigt dies am Beispiel des begehbaren Empfangsbereichs der neuen Kinderklinik.
Die technische Implementierung der in diesem Kapitel aufgeführten Funktionalitäten ist in Kapitel 6.5.1 beschrieben.

Abb. 35: Aktivierung einer zusätzlichen Lichtquelle zur Aufhellung von Innenräumen (Screenshot aus dem Informationssystem, eigene Darstellung)

6.3.3 Erweiterte Steuerung von Ansicht und Navigation

Unterhalb der unter 6.3.2 beschriebenen Basiswerkzeuge finden sich Schaltflächen, die die Standard-Navigationsmöglichkeiten erweitern (Bereich 2 in Abb. 28). Dem Nutzer sind Schaltflächen wie auch Auswahllisten aus vielen Computeranwendungen bekannt. Da die Windows-Standardschaltflächen jedoch sehr viel Platz benötigen, wurden für die graphische Benutzeroberfläche an dieser Stelle kleinere Schaltflächen implementiert. Der Gebrauch entspricht aber den standardmäßig in Windows verfügbaren Schaltflächen. Die Veränderung des Mauszeigers auf einer Schaltfläche zeigt an, dass hier eine Interaktionsmöglichkeit besteht. Zudem wurden diese Schaltflächen so ausgeführt, dass über eine farbliche Veränderung deutlich wird, ob eine Aktivierung erfolgte. Abbildung 36 zeigt die betreffenden Schaltflächen. Die horizontale Anordnung unterstreicht deren Zuordnung zu der Auswahlliste „Ansicht“ bzw. „Navigationsmodus“.

Wählt der Benutzer in der Auswahlliste „Ansicht“ einen Viewpoint aus und verlässt die durch den Viewpoint vorgegebene Position und Ausrichtung, kann er mittels der Schaltfläche „zurück“ wieder an die ursprüngliche Position des ausgewählten Viewpoints zurückgelangen.

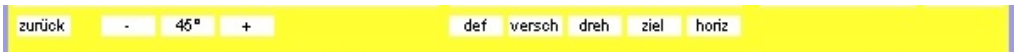

Abb. 36: Schaltflächen zur erweiterten Steuerung von Ansicht und Navigation (Screenshot aus dem Informationssystem, eigene Darstellung)

Mit den Schaltflächen „-“, „45°“, und „+“ kann der Nutzer das Sichtfeld des 3D-Fensters verändern. Für die meisten Viewpoints ist ein Sichtfeld von 45° voreingestellt, da dieses dem menschlichen Sehen am nächsten kommt. Dieses Standard-Sichtfeld wird durch Anklicken der Schaltflächen „-“ bzw. „+“ um jeweils 5° verringert bzw. vergrößert. Die Position des Avatars in der Szene verändert sich dadurch nicht. Der Effekt lässt sich gut mit dem Verwenden unterschiedlicher Objektive an Fotoapparaten vergleichen. Ein Sichtfeld von 45° entspricht dort der Verwendung eines Normalobjektivs mit etwa 50mm Brennweite. Die Verringerung des Sichtfeldes wird durch den Einsatz eines Teleobjektivs erreicht, welches die Dinge näher erscheinen lässt. Eine Brennweite von 135mm entspricht dabei einem Sichtfeld von etwa 15°. Die Verwendung eines Weitwinkelobjektivs etwa mit einer Brennweite von 28mm entspricht einer Vergrößerung des Sichtfelds und einem Wert von rund 65°. Abbildung 37 zeigt dies am Beispiel des für den Botanischen Garten eingestellten Viewpoints.

Abb. 37: Ansicht des Viewpoints „Botanischer Garten“ mit einem Sichtfeld von 15°, 45° bzw. 65° (Screenshots aus dem Informationssystem, eigene Darstellung)

Die Schaltflächen „def “, „versch“ und „dreh“ stehen abkürzend für „default“, „verschieben“ und „drehen“. Diese Funktionen erweitern die in der Auswahlliste „Navigationsmodus“ eingestellte Bewegungsart. Die Standardeinstellung ist „def“. Wird im

Navigationsmodus „gehen“ die Schaltfläche „versch“ aktiviert, so kann über horizontale Ziehbewegungen mit der Maus im 3D-Fenster die aktuelle Position des Avatars seitlich verschoben werden. Über vertikale Ziehbewegungen steht wie in der Grundeinstellung auch das Vorwärts- bzw. Rückwärtsgehen zur Verfügung. Im Navigationsmodus „fliegen“ bewirken horizontale Ziehbewegungen bei aktivierter „versch“-Schaltfläche ebenfalls eine seitliche Verschiebung des Avatars. Vertikale Ziehbewegungen verschieben den Avatar in der Höhe. Die Schaltfläche „dreh“ bewirkt in allen Navigationsmodi dasselbe. Bei vertikalen bzw. horizontalen Ziehbewegungen wird lediglich der Kopf des virtuellen Betrachters bewegt. Die Position des Avatars bleibt unverändert. In Abhängigkeit der Ziehbewegung neigt sich die Blickrichtung nach oben oder unten bzw. nach links oder rechts.
Die Schaltfläche „ziel“ aktiviert eine Funktion, bei der der Benutzer einen bestimmten Punkt bzw. ein bestimmtes Objekt im 3D-Fenster anklickt und dann automatisch dorthin gelangt. Die Position des Avatars wird dabei vom aktuellen Standpunkt an die angeklickte Stelle verlegt. Dies geschieht nicht durch abruptes Wechseln zwischen den beiden Standorten, sondern in animierter Form, bei der sich der Avatar in gerader Linie von der aktuellen Position zur Zielposition bewegt.
Die Schaltfläche „horiz“, was abkürzend für horizontal steht, richtet den Kopf des virtuellen Betrachters auf. Die Blickrichtung wird dabei an einer Ebene parallel zur virtuellen Erdoberfläche ausgerichtet.
Die in diesem Unterkapitel beschriebenen Werkzeuge sind zur Benutzung durch fortgeschrittene Benutzer gedacht und deshalb bewusst als relativ kleine Schaltflächen in einem gesonderten Bereich untergebracht. Die technische Implementierung dieser Funktionen ist in Kapitel 6.5.2 beschrieben.

6.3.4 Steuerung der Szeneninhalte

Im in der Abbildung 28 mit 4 bezeichneten Bereich stellen insgesamt vier Registerkarten die zweite Kernfunktionalität des Systems bereit. Mithilfe dieser Registerkarten werden bestimmte, innerhalb der Benutzeroberfläche zusammengehörige Elemente zusammengefasst und übereinander angeordnet. Das Konzept der Registerkarten ist dem Benutzer ebenfalls aus den meisten Standard-Computeranwendungen vertraut. Die Bedienung ist intuitiv. Das Anklicken eines Reiters in der Registerleiste rückt die dazugehörige Registerkarte in den Vordergrund.

Die Registerkarte „Layer 3D“ hält die Möglichkeit bereit, bestimmte weitere dreidimensionale Inhalte in die 3D-Szene einzublenden. Innerhalb dieser Registerkarte sind dazu so genannte Checkboxen sowie beschreibender Text aufgelistet. Solche Checkboxen sind ebenfalls ein geläufiges Konzept auf Webseiten oder in Standard-Computeranwendungen. In Abhängigkeit des Zustands einer Checkbox, d.h. ob ein Haken gesetzt ist oder nicht, wird der damit verknüpfte dreidimensionale Inhalt eingeblendet oder ausgeblendet.

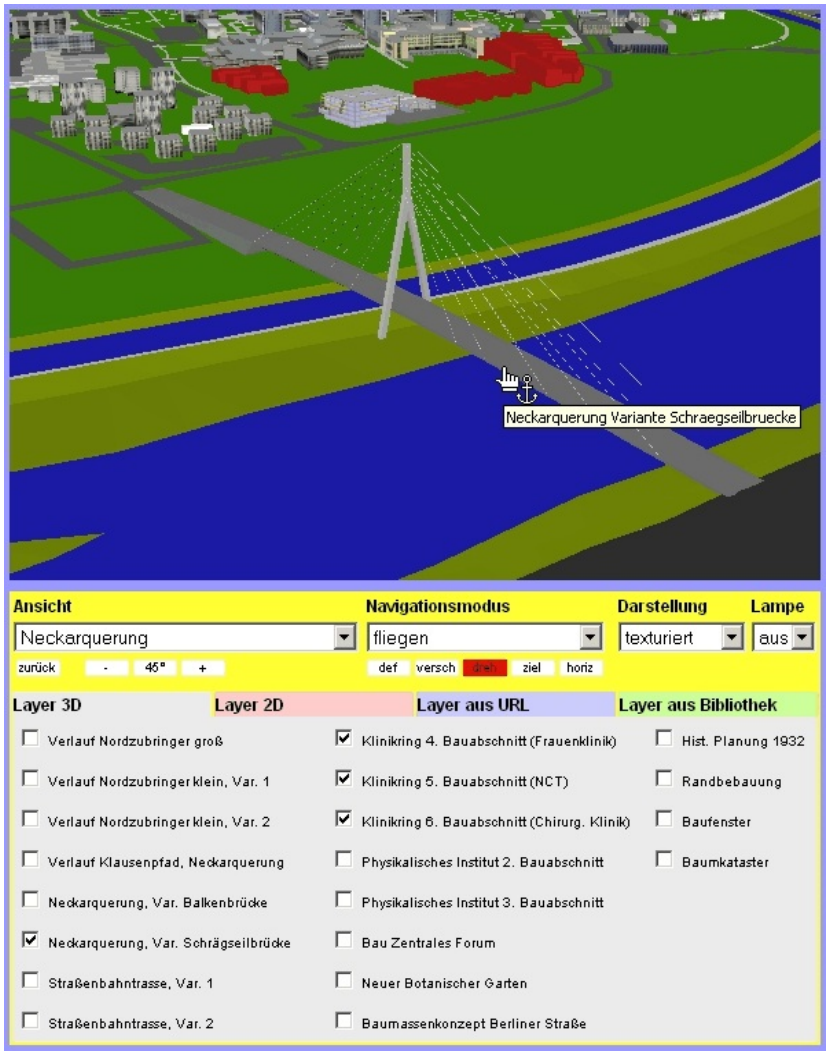

Abb. 38: Ansicht des Viewpoints „Neckarquerung“ mit zusätzlich eingeblendeten Inhalten auf der Registerkarte „Layer 3D“ (Screenshot aus dem Informationssystem, eigene Darstellung)

Abbildung 38 zeigt die Ansicht des Viewpoints „Neckarquerung“ in der 3D-Szene. Zusätzlich zu dem dreidimensionalen Modell, welches den Baubestand des Universitäts-Campus zeigt, sind vier weitere Ebenen aktiviert. Dies sind die Schrägseilvariante einer möglichen Neckarquerung sowie im Hintergrund die zukünftigen Bauabschnitte des Heidelberger Klinikrings. Auch für diese Objekte kann der Nutzer durch Anklicken zusätzliche Informationen abfragen.

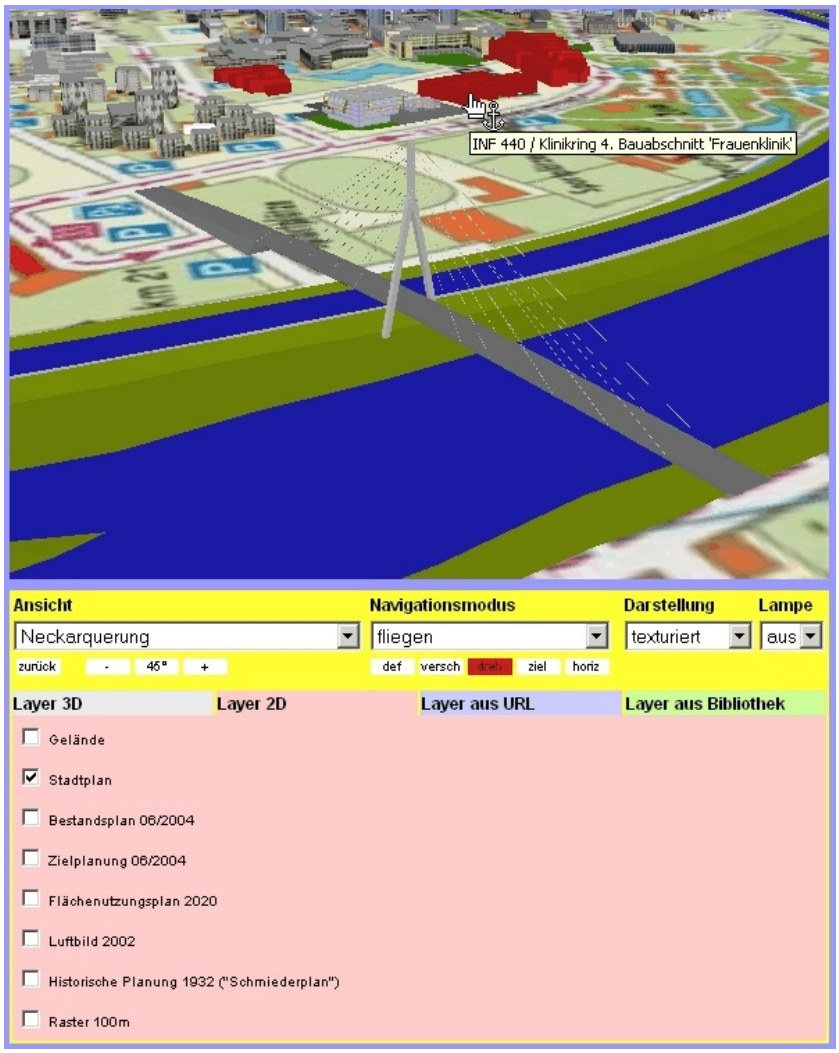

Abb. 39: Ansicht des Viewpoints „Neckarquerung“ mit zusätzlich eingeblendeten Inhalten auf der Registerkarte „Layer 2D“ (Screenshot aus dem Informationssystem, eigene Darstellung)

Die Registerkarte „Layer 2D“ gleicht prinzipiell der Karte „Layer 3D“. Lediglich die Inhalte, die über die Checkboxen dieser Registerkarte in der Szene ein- bzw. ausgeblendet werden können, gehören zu einer anderen Kategorie. Hier werden keine dreidimensionalen Objekte, sondern eher zweidimensionale Elemente in der Szene aktiviert bzw. deaktiviert. Die Einschränkung „eher“ bedeutet, dass einige der Inhalte auf dieser Registerkarte durch ihre sehr große Ausdehnung bei im Vergleich dazu sehr geringen Höhenunterschieden vom Betrachter als zweidimensional wahr genommen werden. Der in der Grundeinstellung des Systems aktivierte Layer „Gelände“ ist ein Beispiel dafür. Dieser Layer enthält Straßen, Gehwege und Grünflächen, die sehr wohl Höhenunterschiede von mehreren Dezimetern aufweisen. Das für dieses Projekt eigens digitalisierte Neckarbett sowie die auf eigenen Erhebungen beruhenden Geländemodelle um die Medizinische Klinik und neue Kinderklinik sind nicht Teil des Layers „Gelände“. Diese Geometrien sind permanent aktiviert, weil es sich bei ihnen um charakteristische Reliefunterschiede im Bereich des Neuenheimer Feldes handelt. Abbildung 39 zeigt denselben Bildausschnitt sowie denselben Viewpoint wie die vorangegangene Abbildung. Statt der Registerkarte „Layer 3D“ ist die Karte „Layer 2D“ ausgewählt. Der Layer „Gelände“ wurde deaktiviert und stattdessen der Layer „Stadtplan“ (Quelle: VERMESSUNGSAMT HEIDELBERG 2003) für die Anzeige ausgewählt. Diese neue Kombination der Inhalte transportiert entsprechend andere Informationen.
Die Funktionalität der Registerkarte „Layer aus URL“ unterscheidet sich grundlegend von den ersten beiden. Während diese die Möglichkeit bieten vordefinierte Inhalte ein- oder auszublenden, können über die Funktion in der Registerkarte „Layer aus URL“ benutzerdefinierte Inhalte von beliebigen Stellen des Internets, eines Intranets oder von der lokalen Festplatte des Benutzers in die aktuelle Szenenansicht geladen werden. Der Benutzer gibt dabei den Pfad, d.h. den Ort, an dem sich die Datei befindet, die er laden möchte, sowie deren vollständigen Dateinamen ein. Diese Angabe wird auch als Uniform Ressource Locator (URL) bezeichnet. Ein Begriff aus der Informatik, der eine physische oder abstrakte Ressource definiert. Nach Angabe dieses URL muss durch Anklicken die Schaltfläche „Laden“ aktiviert werden, um den Vorgang zu starten. Die Inhalte, die auf diese Weise in die Szene geladen werden sollen, müssen dabei als VRML-Datei vorliegen. Darüber hinaus ist es notwendig, dass sie die in Kapitel 4.1.1 bzw. 5.3.3 aufgestellten Anforderungen an eine Georeferenzierung erfüllen. Abbildung 40 zeigt im 3D-Fenster einen Teilbereich der Berliner Straße sowie die aktivierte Registerkarte „Layer aus URL“. Als URL wurde eine Datei angegeben, die sich auf einem Internetserver befindet. Es handelt sich bei der Datei um ein 3D-Modell einer

fiktiven Variante eines Baumassenkonzepts für die Berliner Straße. Die Geometrie, die diese Datei enthält, wird in der Szene als rotes Klötzchenmodell dargestellt. Über die Registerkarte „Layer 2D“ wurden für die Szene ein Luftbild des Neuenheimer Feldes (Quelle: VERMESSUNGSAMT HEIDELBERG 2003) sowie ein Raster mit einem Zellenabstand von 100 m aktiviert, um die Maßstabsverhältnisse zu verdeutlichen.

Abb. 40: Ansicht des Viewpoints „Berliner Straße“ mit benutzerdefiniert eingeblendeten Inhalten auf der Registerkarte „Layer aus URL“ (Screenshot aus dem Informationssystem, eigene Darstellung)

Die vierte Registerkarte „Layer aus Bibliothek“ dient zwar auch wie die drei anderen Karten dazu, die Inhalte der Szene zu verändern, unterscheidet sich aber in ihrer Funktionalität ebenso deutlich von den ersten dreien wie die zuletzt beschriebene von den ersten beiden.

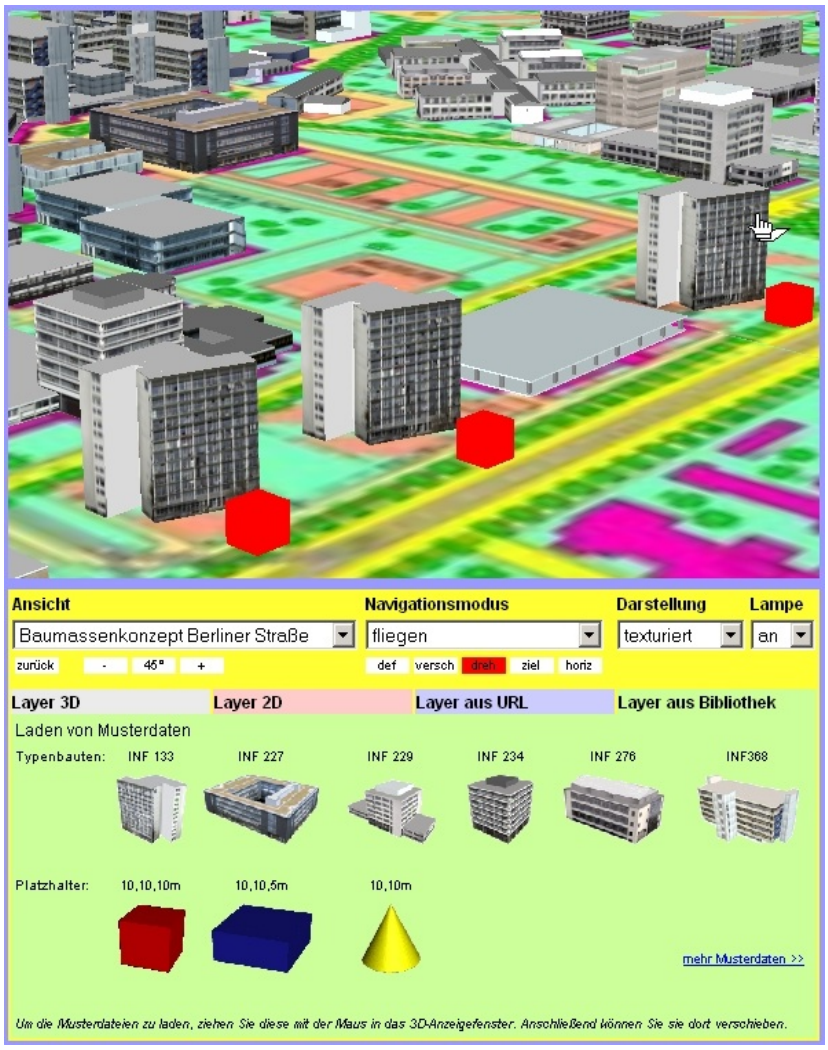

Abb. 41: Ansicht des Viewpoints „Berliner Straße“ mit aktiv vom Benutzer eingebrachten und frei positionierbaren Musterdaten (Screenshot aus dem Informationssystem, eigene Darstellung)

Diese Registerkarte enthält keine für den Benutzer direkt erkennbaren Standardinteraktionselemente wie dies hinsichtlich der Checkboxen, des Eingabefeldes oder der Schaltfläche auf den ersten drei Karten der Fall ist. Stattdessen zeigt die Registerkarte

„Layer aus Bibliothek“ Abbildungen charakteristischer Typenbauten (Kap. 3.2.2) des Universitätscampus. Neben diesen Typenbauten sind mehrere geometrische Standardelemente mit definierten Ausmaßen abgebildet. Hinter diesen Abbildungen stehen einzelne Dateien, die die in der Abbildung gezeigten Objekte enthalten. Mit der dem Benutzer aus anderen Computer-Anwendungen vertrauten Drag & Drop Methode können diese Objekte der aktuellen Szene hinzugefügt werden. Durch Ziehen einer Abbildung in das 3D-Fenster wird das entsprechende Objekt exakt an der Stelle der Szene erzeugt, an welcher es fallen gelassen wird. Die Objekte, die in dieser Bibliothek von Musterdaten zur Auswahl stehen, werden über die Drag & Drop Methode nicht unabänderlich in der Szene verankert, sondern der Benutzer kann sie auch im Nachhinein „anfassen“ und beliebig in der Szene positionieren. Über einen Verweis innerhalb der Registerkarte hat der Nutzer Zugriff auf weitere Typenbauten und geometrische Platzhalter. Abbildung 41 zeigt im 3D-Fenster einen Teilausschnitt des Campus entlang der Berliner Straße. Der Szene wurden drei standardisierte Studentenwohnheime hinzugefügt und links bzw. rechts neben der Parkpalette positioniert. Zusätzlich wurden drei geometrische Platzhalter, Würfel mit einer Kantenlänge von 10m, vor diesen Wohnheimen platziert. Als weitere Informationsebene wurde in der Registerkarte „Layer 2D“ statt des Geländes die Zielplanung für das Untersuchungsgebiet aktiviert (Quelle: UNIVERSITÄTSBAUAMT HEIDELBERG 2004b).
Die technische Implementierung der in diesem Kapitel aufgeführten Funktionen ist in den Kapiteln 6.5.4, 6.5.5 und 6.5.6 beschrieben.

6.3.5 Positionsanzeige und absolute Positionierung des Avatars

Die Bereiche 5 und 6 in Abbildung 28 dienen in erster Linie dazu, dem Benutzer die aktuelle Position des Avatars in der Szene zu zeigen und ihm so die Orientierung zu erleichtern. Dies geschieht zum einen grafisch und zum anderen auf der Basis von Zahlen. Der Bereich 5 wird dabei von einem genordeten Übersichtsplan (QUELLE: UNIVERSITÄTSBAUAMT HEIDELBERG 2004a) eingenommen, der das Neuenheimer Feld mit seinen Einrichtungen darstellt. Dieser Plan steht auch an diversen Punkten des realen Campus´ auf großflächigen Schautafeln zur Orientierung zur Verfügung. Dieser Plan wird von verschiedenen Elementen der graphischen Benutzeroberfläche überlagert. Dies sind ein roter Pfeil sowie zwei sich kreuzende gelbe Linien. Die Position des roten Pfeils auf der Karte zeigt die aktuelle Betrachterposition des Avatars in der Szene an. Die Ausrichtung des Pfeils zeigt die jeweilige Blickrichtung des Avatars an. Die

gelben Linien dienen dem Benutzer zur schnelleren optischen Erfassung des Pfeils. Sie bilden zudem Achsen desselben Rechts- bzw. Hochwerts innerhalb des Gauß-Krüger Koordinatensystems. Da der Übersichtsplan nicht alle Bereiche abdeckt, die der Avatar in der Szene einnehmen kann, erfüllen die gelben Linien überdies den Zweck, den Sektor zu bezeichnen, in dem sich der Avatar in einem solchen Fall befindet. Die Linien zeigen solange die Achse an, auf der sich der Betrachter aufhält, wie sich der Avatar in einem der vier bündig an den Übersichtsplan anschließenden Sektoren befindet. Alternativ dazu ließe sich relativ einfach eine Zoom- oder Panfunktion realisieren, die den Planausschnitt in entsprechender Größe anzeigt bzw. diesen verschiebt. Im Regelfall wird durch den Übersichtsplan jedoch ein ausreichend großer Aktionsradius des Avatars in der Szene abgedeckt. Die Positionierung des Pfeils und der Linien erfolgt zeitgleich mit der Bewegung des Avatars in der 3D-Szene. Abbildung 42 zeigt den entsprechenden Bereich der Benutzeroberfläche mit den beschriebenen Elementen. Der Bereich 6 wird eingenommen von sechs beschrifteten Feldern. Die ersten beiden Felder geben die Gauß-Krüger-Koordinaten der Position des Avatars als Rechts- und Hochwert an. Die mittleren zwei Felder geben die Höhe des Avatars über Normalnull bzw. über Grund an. Letzterer Wert wird dabei auf eine Kommastelle genau ausgegeben, um den Benutzer im Navigationsmodus „gehen“ über die Aughöhe des Avatars von 1,6m zu informieren. Dies ist die Augpunkthöhe eines durchschnittlich großen Menschen. Das Feld mit der Beschriftung Azimut gibt den genauen Wert des Nordwinkels des Avatars an. Dies geschieht in Winkelgrad. Das Ausgabefeld mit der Bezeichnung Sichtfeld informiert den Benutzer über die Breite des Sichtfelds des aktiven Viewpoints, d.h. der aktuellen Ansicht im 3D-Fenster. Auch dieser Wert wird in Winkelgrad ausgegeben. Alle Werte werden ebenfalls in Echtzeit aktualisiert. Abbildung 42 zeigt die beschriebenen Elemente.

Zusätzlich zu der beschriebenen Funktionalität, die den Benutzer nur in rezipierender Weise fordert, erlaubt die Übersichtskarte auch eine Interaktion. Durch Klicken auf den Plan wird die aktuelle Position des Betrachters an die angeklickte Stelle auf der Karte verlegt. Der Mauszeiger erscheint für diese Funktion als Fadenkreuz (Abb. 42, links oberhalb des roten Pfeils). Dies geschieht nicht in animierter Art und Weise wie bei der Selektion vorgegebener Viewpoints in Kapitel 6.3.2 beschrieben, sondern abrupt. Diese Funktion ist wie die Schaltflächen in 6.3.3 für den versierten Benutzer gedacht. Hiermit wird ein extrem schnelles und vom dreidimensionalen Kontext vollkommen losgelöstes Positionieren des Avatars ermöglicht. Mit der Aktivierung dieser Funktion wird in der Auswahlliste „Ansicht“ (vgl. Kap. 6.3.2) ein eigener Eintrag mit

der Bezeichnung „benutzerdefiniert“ aktiv. In diesem Eintrag werden die Koordinaten des mithilfe dieser Methode positionierten Avatars abgespeichert. So kann die Ansicht, die über diese absolute Positionierung erzeugt wurde, solange wieder aufgerufen werden, wie sie nicht mit anderen Koordinaten durch erneutes Ausführen der Funktion überschrieben wird. Da der Übersichtsplan ein zweidimensionales Medium darstellt, wird für die Höhe des erzeugten Betrachterstandpunktes immer der Wert 1,6m, also Aughöhe über Grund aufgerufen. Die Ausrichtung des Avatars erfolgt mit einem auf 360° eingestellten Standardwert. Über eine Ziehbewegung auf der Karte ließe sich jedoch auch eine benutzerspezifische Generierung der Blickrichtung erreichen.
Die technische Implementierung der oben geschilderten Funktionen ist in den Kapiteln 6.5.3 und 6.5.8 beschrieben.

Abb. 42: Bereich der Benutzeroberfläche, der über Position und Blickrichtung des Avatars informiert sowie die absolute Positionierung des Avatars erlaubt (Screenshot aus dem Informationssystem, eigene Darstellung)

6.4 Dokumentstruktur

Der Funktionsablauf und die Inhalte des Informationssystems werden maßgeblich von zwei auf dem INF3D-Server liegenden Dokumenten bestimmt bzw. verwaltet. Dies ist zum einen das HTML-Dokument index.html, welches die graphische Benutzeroberfläche enthält und einen Knotenpunkt für die Funktionalitäten der Informationsplattform bildet. Zum anderen ist dies das VRML-Dokument root.wrz, welches die dreidimensionalen Inhalte einbindet und den zweiten Knotenpunkt für die Funktionalitäten des Systems bildet.
Abbildung 43 zeigt, welche Dokumente auf dem INF3D-Server gespeichert sind und wie einzelne Dokumente bzw. Dokumentgruppen miteinander verknüpft sind. Insgesamt werden rund 800 Dokumente bzw. Dateien auf dem Server gespeichert und von den beiden Hauptdokumenten, index.html und root.wrz, verwaltet. Der Großteil dieser Dateien sind Geometrien einzelner Gebäude sowie die mit diesen verknüpften Fotos der Gebäudefassaden.

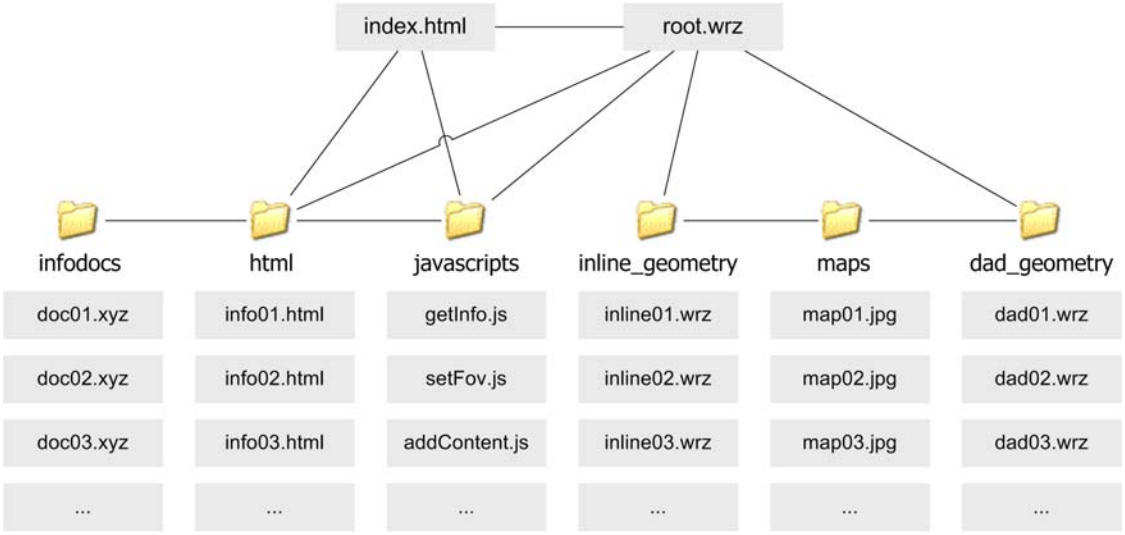

Abb. 43: Dokumente des Informationssystems und deren Verknüpfung (eigene Darstellung)

Mit dem Aufruf der Datei index.html (Abb. 43) durch den Internetbrowser des Clientrechners werden alle anderen Dokumente erschlossen. Zuerst wird das VRML-Dokument root.wrz über die entsprechenden ActiveX-Komponenten (Kap. 6.1.7) eingebunden. Dieses Dokument enthält keinerlei Geometrien dreidimensionaler Objekte, sondern lediglich Verweise, wo Dokumente mit den entsprechenden Geometrien zu

finden sind. Die Geometrien des Gebäudebestandes sowie der historischen und zukünftigen Planung sind als VRML-Dokumente in dem Ordner inline_geometry abgelegt. Die Geometrien, die der Benutzer über Drag & Drop aktiv in die Szene einbringen kann (Kap. 6.3.4), sind als VRML-Dokumente in dem Ordner dad_geometry gespeichert. Die VRML-Dokumente dieser beiden Ordner enthalten wiederum Verweise auf den Ordner maps, der die mit der Geometrie verknüpften Texturen enthält. Diese Gliederung ermöglicht es, den Verwaltungsaufwand der Inhalte der 3D-Szene auf ein einziges zentrales Dokument, nämlich root.wrz, zu fokussieren. Andererseits gestattet es diese Dokumentordnung aber auch, einzelne Gebäudegeometrien dezentral abzulegen, so dass deren Pflege bzw. Aktualisierung in der Praxis von Dritten übernommen werden kann, was angesichts der Fülle der Inhalte vorteilhaft ist. Diese Eigenschaften waren im Konzept gefordert. Der Ordner javascripts, auf dessen Dokumente sowohl index.html und root.wrz zugreifen, enthält die für die Steuerung einzelner Systemfunktionen notwendigen JavaScipt-Programme. Es wäre ebenso möglich, diese Programme direkt in das HTML-Dokument einzubinden. Durch die separate Speicherung werden die JavaScript-Programme auch für andere HTML-Dokumente aufrufbar. Außerdem erleichtert die getrennte Speicherung die Pflege bzw. Erweiterung des Gesamtsystems.
Der Ordner html fasst die HTML-Dokumente zusammen, die bei der Abfrage einzelner Objektinformationen in der 3D-Szene (Kap. 6.3.1) aufgerufen werden. Diese können ihrerseits mit Dokumenten verschiedener Art verknüpft sein, die weitere Informationen zu einzelnen Objekten liefern.
In den nachfolgenden Kapiteln wird die Struktur der beiden wichtigsten Dokumente, index.html und root.wrz, erläutert. Der Aufbau und die Funktionsweise der JavaScript-Programme werden in Kapitel 6.5 detailliert beschrieben.

6.4.1 HTML-Dokument index.html

Abbildung 44 zeigt den Knotenbaum des Document Object Models, welcher der zentralen HTML-Seite zu Grunde liegt. Der Aufbau dieser Baumstruktur wird nachfolgend erläutert. Bei denjenigen Elementen, die eine Rolle für die graphische Gestaltung der Benutzeroberfläche spielen, wird die entsprechende Beziehung hergestellt.
Innerhalb der genannten Abbildung ist der Typ eines Knotens in jedem Feld in Großbuchstaben notiert. Die dokumentweit eindeutige ID bzw. der Name eines Knotens ist in kursiven Kleinbuchstaben notiert. Unterbrochene Linien innerhalb der Baumstruktur kennzeichnen, dass mehrere Knoten dieses Typs auf derselben Hierarchieebene exis-

tieren. Um die schematische Darstellung übersichtlicher zu gestalten, wird in diesem Fall jeweils nur ein Knoten desselben Typs in der Baumstruktur dargestellt. Abgebildet sind zudem nur diejenigen Knoten, welche für den Funktionsablauf bzw. die schematische Gliederung der Benutzeroberfläche wesentlich sind. Der tatsächliche DOM-Baum ist wesentlich komplexer, wie Abbildung 45 illustriert, in der das zum Teil aufgeklappte DIV-Element *control1* dargestellt wird. Neben der Baumstruktur, die das hierarchische Verhältnis der Knoten zueinander beschreibt, kennzeichnen in Abbildung 44 Pfeile den Aufruf von Javascript-Funktionen bzw. die Übergabe von Werten sowie Lese- und Schreibzugriffe auf bestimmte Bestandteile des Dokuments.
Im Weiteren werden die einzelnen Elemente der Abbildung 44 beschrieben. Die Typbezeichnung eines Elements sowie gegebenenfalls dessen ID bzw. Name werden in derselben Schreibweise wie in der Abbildung gebraucht, um deutlich zu machen, auf welche Knoten des Dokuments Bezug genommen wird. Synonym zum Ausdruck Knoten wird der Terminus Element gebraucht. Wie einzelne Elemente den Aufruf von JavaScript-Programmen steuern und wie dadurch Inhalte dynamisch verändert werden, wird in den darauffolgenden Kapiteln detailliert beschrieben.
HTML-Dokumente bestehen wie in Kapitel 6.1.2 erläutert im Wesentlichen aus zwei Teilen; dem Head und dem Body. Der Head (HEAD) des Dokumentes index.html enthält neben den üblichen Elementen wie Meta-Angaben (META) und Namen (TITLE) des Dokuments so genannte Style Sheets (STYLE). Das für die Ausführungen in den nachfolgenden Kapiteln wichtigste Element ist der Script-Knoten (SCRIPT). Mit diesem wird ein Bereich im Dokument definiert, welcher Anweisungen in einer Script-Sprache enthält, bzw. durch den externe Scripte in das Dokument eingebunden werden. Das grau unterlegte, mit dem Script-Knoten verbundene Feld listet in alphabetischer Reihenfolge diejenigen JavaScript-Programme auf, welche speziell für die Steuerung der Funktionsmodule programmiert wurden. Die genaue Funktionsweise wird in Kapitel 6.5 erläutert. Neben den in Abbildung 44 explizit aufgelisteten Scripte wurden weitere programmiert, welche für die Darstellung der graphischen Benutzeroberfläche notwendig sind. Auf diese wird nicht gesondert eingegangen, da es sich im Wesentlichen um Standardprozeduren handelt, die in der entsprechenden Literatur (vgl. Kap. 6.1) ausreichend diskutiert werden. Der Body (BODY) enthält auf der ersten Hierarchieebene zehn so genannte Bereichsknoten (DIV). Diese Elemente lassen sich den in Abbildung 28 dargestellten Bereichen der graphischen Benutzeroberfläche zuordnen. Im Folgenden wird näher auf Aufbau und Funktion der einzelnen Bereichsknoten eingegangen.

Der DIV-Knoten *vrml* enthält das OBJECT-Element *vrmlscene*. Über den OBJECT-Knoten werden Dateien oder Datentypen, die sich außerhalb des HTML-Dokuments befinden, eingebunden. Im vorliegenden Fall wird ein ActiveX-Control (Kap. 6.1.7) eingebunden. Über spezifische Attribute wird definiert, dass das Cortona ActiveX-Control eingebunden werden soll. Darüber hinaus wird festgelegt, dass Cortona automatisch von einer bestimmten Internetadresse geladen und ausgeführt werden soll, falls es auf dem Client-Rechner noch nicht verfügbar sein sollte. Der Aufruf des ActiveX-Controls erfolgt unter Übergabe bestimmter Parameter. Über diese wird unter anderem definiert, dass die Datei root.wrz in Cortona geöffnet werden soll. Der eingehende Pfeil kennzeichnet, dass auf das OBJECT-Element mithilfe verschiedener JavaScript-Programme zugegriffen wird. Das OBJECT-Element gibt seinerseits Werte aus, die von den JavaScript-Programmen weiterverarbeitet werden. Dies wird durch den ausgehenden Pfeil verdeutlicht. Die für den Benutzer in der graphischen Benutzeroberfläche sichtbaren Elemente des DIV-Knotens *vrml* und der ihm untergeordneten Knoten entsprechen dem mit 1) bezeichneten Bereich in Abbildung 28.
Der DIV-Knoten *control1* ist einer von insgesamt vier DIV-Knoten, die dem Nutzer Interaktionsmöglichkeiten mit dem System zur Verfügung stellen. Dem DIV-Element *control1* ist ein Formular-Knoten (FORM) mit der ID *form1* untergeordnet. Dieser Formularknoten ist wie auch die weiteren Formularknoten in den anderen Bereichsknoten nicht zwingend notwendig, da die Formulardaten in der momentanen Ausbaustufe des Systems nicht an den Server übertragen, sondern vollständig durch den Client-Rechner verarbeitet werden. Im Hinblick auf den weiteren Ausbau des Systems, eventuell im Zusammenspiel mit der AJAX-Technik (GAMPERL 2007) erschien es jedoch zweckmäßig, sämtliche Bereiche, in denen Benutzereingaben stattfinden, bereits frühzeitig als Formularknoten zu implementieren. Das Formular *form1* enthält seinerseits mehrere Auswahlelemente (SELECT) mit den IDs *list1* bis *list4*. Diese Knoten sind als so genannte Auswahllisten umgesetzt, welche als hierarchisch niedrigere Knoten die Options-Elemente (OPTION) enthalten. Die in der graphischen Benutzeroberfläche sichtbaren Elemente des DIV-Elements *control1* und seiner nachgeordneten Knoten sind in Abbildung 28 unter Punkt 2 zu sehen. Die SELECT-Elemente *list1* bis *list4* entsprechen dabei den Auswahllisten „Ansicht“, „Navigationsmodus“ usw., während die einzelnen Listeneinträge wie „Draufsicht“, „kein“ usw. den OPTION-Elementen der einzelnen SELECT-Knoten entsprechen. Das DIV-Element *control1* enthält weitere für den Nutzer nicht sichtbare Elemente, die für die interne Steuerung der JavaScript-Programme verantwortlich sind (vgl. Kap. 6.5).

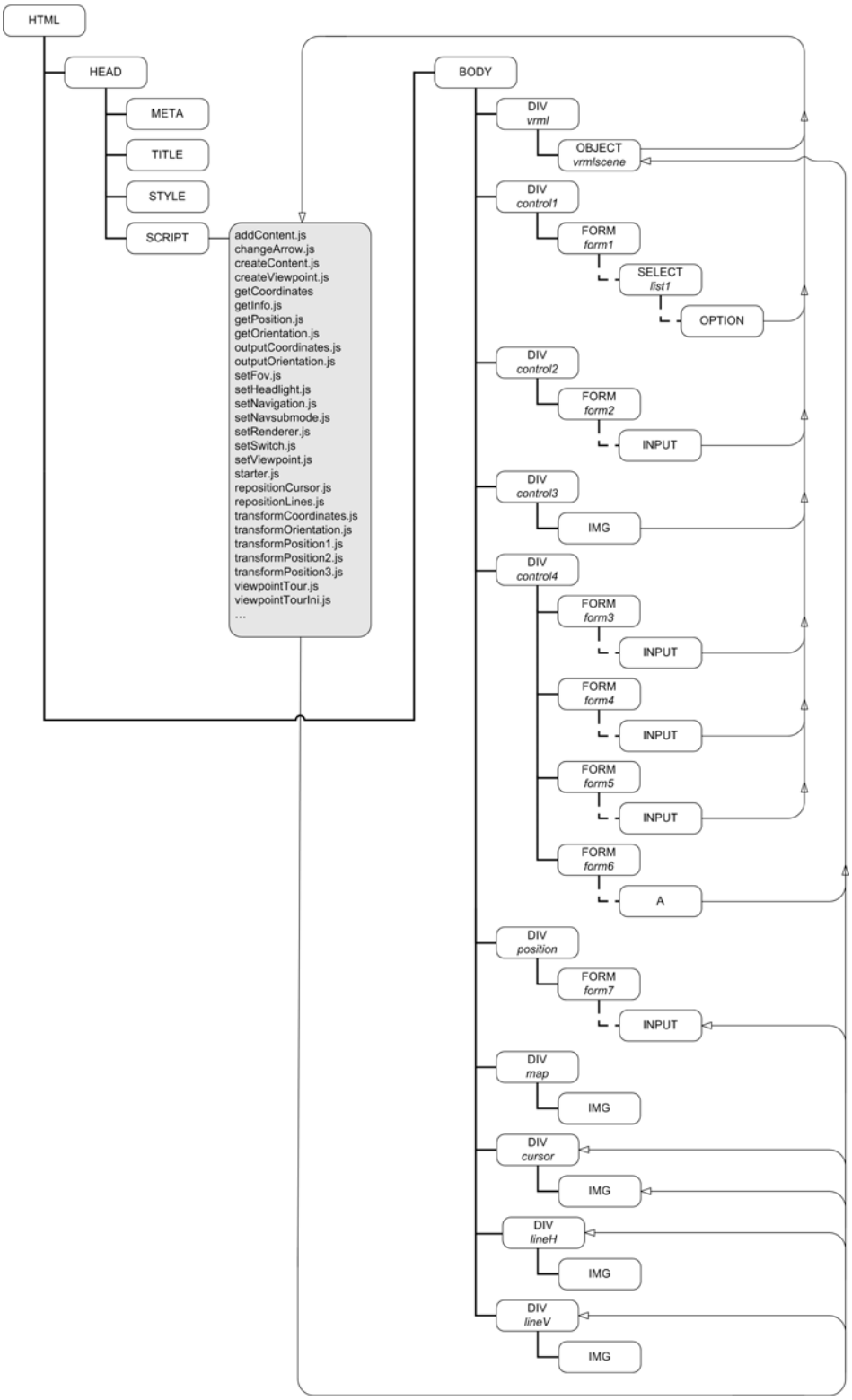

Abb. 44: DOM-Baum der Datei index.html (eigener Entwurf und Darstellung)

Der DIV-Knoten *control2* enthält ebenfalls zunächst ein Formular-Element mit der ID *form2*. Diesem sind mehrere Eingabe-Knoten (INPUT) untergeordnet. Diese Eingabe-Elemente sind als Schaltflächen ausgeführt, die den Aufruf von JavaScript-Programmen kontrollieren. Sie entsprechen den Schaltflächen, die in Abbildung 28 unter Punkt 3 markiert sind.
Das DIV-Element *control3* enthält einen untergeordneten Image-Knoten (IMG). In der graphischen Schnittstelle ist dieses Element für den Benutzer nicht sichtbar. Es stellt die Grundlage für das in Kapitel 6.5.3 beschriebene Funktionsmodul zur absoluten Positionierung des Avatars dar.

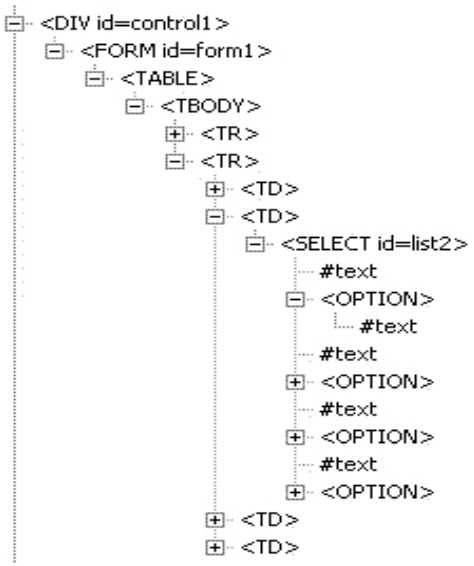

Abb. 45: Abschnitt des DOM-Baums für das DIV-Element control1 (eigene Darstellung)

Der DIV-Knoten *control4* stellt die in den Kapiteln 6.5.4, 6.5.5 und 6.5.6 beschriebene Schnittstelle zur Steuerung der Szeneninhalte bereit. Ihm sind die Formular-Elemente *form3* bis *form6* nachgeordnet. Die für den Nutzer sichtbaren Bestandteile dieser Formulare entsprechen den in Abbildung 28 unter Punkt 4 gekennzeichneten Registerkarten „Layer 3D“, „Layer 2D“, „Layer aus URL“ und „Layer aus Bibliothek“. Die ersten drei Formularknoten enthalten jeweils mehrere Eingabe-Elemente des Typs INPUT. Während die Formularknoten *form3* und *form4* Checkboxen zur Verfügung stellen, steht dem Benutzer im Formularelement *form5* ein einzeiliges Feld für die Eingabe von Dateipfaden bzw. URLs zur Verfügung. Der vierte Formularknoten *form6* enthält keine Eingabe-Elemente des Typs INPUT, sondern eine Reihe von Verweisen, die durch

den Knotentyp A für Anchor (vgl. Kap. 6.1.2) gekennzeichnet sind. Auf die mit den Elementen des DIV-Knotens *control4* verbundene interne Kontrolle der JavaScript-Programme wird in Kapitel 6.5 eingegangen.
Das DIV-Element *position* enthält in einem weiteren Formular-Knoten *form7* wiederum Eingabe-Felder des Typs INPUT. Diese werden allerdings nicht zur Eingabe, sondern als Ausgabefelder genutzt (vgl. Kap. 6.1.2 bzw. 6.5). Die sichtbaren Bestandteile dieses Knotens sind in Abbildung 28 unter Punkt 6 dargestellt.
Der DIV-Knoten *map* enthält wie auch der DIV-Knoten *control3* ein untergeordnetes Image-Element. Dieses ist für den Benutzer sichtbar und stellt die in Abbildung 28 unter Punkt 5 dargestellte Karte dar.
Die DIV-Elemente *cursor*, *lineH* und *lineV* sind gleich aufgebaut und enthalten ebenfalls ein für den Nutzer sichtbares Image-Element (IMG). Diese drei DIV-Elemente liegen bezüglich ihrer Sichtbarkeit für den Benutzer über dem DIV-Knoten *map*, werden aber im Unterschied zu diesem dynamisch über JavaScript-Prozeduren angesteuert (vgl. Kap. 6.5.8). In Abbildung 28 sind diese DIV-Elemente ebenfalls unter Punkt 5 zu sehen. Es handelt sich um den roten Richtungspfeil sowie die beiden gelben horizontal bzw. vertikal verlaufenden Linien.

6.4.2 VRML-Dokument root.wrz

Abbildung 46 zeigt die Anordnung der einzelnen VRML-Knoten anhand des Szenegraphen, der der zentralen VRML-Datei zu Grunde liegt. Der Aufbau sowie die einzelnen Knoten dieses Szenegraphen werden nachfolgend erläutert.
Jeder Kasten repräsentiert einen Knoten. Im oberen Bereich jedes Kastens ist der Knotentyp in Normalschrift beginnend mit einen Großbuchstaben notiert. Darunter steht der dokumentweit eindeutige Knotenname in kursiver Schrift, welcher ebenfalls mit einem Großbuchstaben beginnt. Im unteren Bereich eines Kastens sind diejenigen Felder des Knotens dargestellt, die wesentlichen Einfluss auf seine Funktionalität haben. Pro Zeile ist jeweils ein Feld aufgeführt. Als erstes ist in Kleinbuchstaben die Zugriffsart des Feldes und darauffolgend der Feldname angegeben. Unterbrochene Linien innerhalb des Szenegraphen kennzeichnen, dass mehrere Knoten dieses Typs auf derselben Hierarchieebene existieren. Um die schematische Darstellung übersichtlicher zu gestalten, wird in diesem Fall jeweils nur ein Knoten im Szenegraphen dargestellt. Abgebildet sind zudem nur diejenigen Knoten, welche für die Darstellung der dreidimensionalen Szene bzw. den internen Programmablauf wesentlich sind. Der tatsächli-

che Szenegraph ist deutlich umfangreicher. Neben der Baumstruktur des Szenegraphen, die das hierarchische Verhältnis der Knoten zueinander beschreibt, kennzeichnen Pfeile den Aufruf von Javascript-Funktionen, VRML-Routen oder die Übergabe von Werten sowie Lese- und Schreibzugriffe auf bestimmte Bestandteile des Dokuments.
Im Weiteren werden ausgewählte Elemente des Dokuments beschrieben. Die Typbezeichnung eines Knotens sowie dessen Name und relevante Felder werden in derselben Schreibweise wie in der Abbildung gebraucht, um deutlich zu machen, auf welche Knoten Bezug genommen wird. Synonym zum Ausdruck Knoten wird der Terminus Element gebraucht. Wie die einzelnen Elemente durch den Aufruf von JavaScript- bzw. VRMLScript-Programmen oder VRML-Routen dynamisch verändert werden, wird in den folgenden Kapiteln beschrieben.
Der Knoten NavigationInfo steuert über drei Felder der Zugriffsart exposedfield die Eigenschaften des Avatars, den Zustand einer zusätzlichen Lichtquelle sowie die Art der Navigation in der Szene. Das Feld avatarSize ist dabei mit drei Werten belegt. Über den ersten Wert wird definiert, wie groß der Mindestabstand zu geometrischen Objekten sein soll, bevor es zu einer Kollision mit dem Avatar kommt. Der zweite Wert gibt die Höhe der Kamera über dem Boden an, wenn die Navigationsart „gehen“ gewählt wurde. Dieser Wert beträgt standardmäßig 1,6m; die schon erwähnte durchschnittliche Augpunkthöhe. Über den dritten Wert kann definiert werden, wie hoch ein Hindernis maximal sein darf, damit der Avatar dies noch übersteigen kann. Dieser Wert ist beispielsweise dann wichtig, wenn dem Avatar die Möglichkeit gegeben werden soll, Treppen zu bewältigen. Das Feld headlight ist verantwortlich für die Aktivierung einer zusätzlichen Lichtquelle, welche Licht parallel zur Blickrichtung, ähnlich einer Helmlampe, aussendet. Diese an den Avatar gebundene Lichtquelle ist wichtig, wenn Innenräume von Gebäuden betreten werden, die über keine eigene Beleuchtung verfügen. Über das Feld type kann die in einer Szene gültige Navigationsart definiert werden. Während das Feld avatarSize mit spezifischen Werten belegt wurde, wurde für die beiden anderen Felder jeweils deren Standard-Einstellung übernommen. Die graphische Schnittstelle bietet dem Benutzer die Möglichkeit sowohl den Status der zusätzlichen Lichtquelle wie auch die Auswahl eines bestimmten Navigationsmodus zu bestimmen. Hierbei wird allerdings nicht dynamisch auf die entsprechenden Felder des NavigationInfo-Elements zugegriffen, sondern eine spezielle Möglichkeit des Cortona ActiveX Controls genutzt (vgl. Kap. 6.5.1).

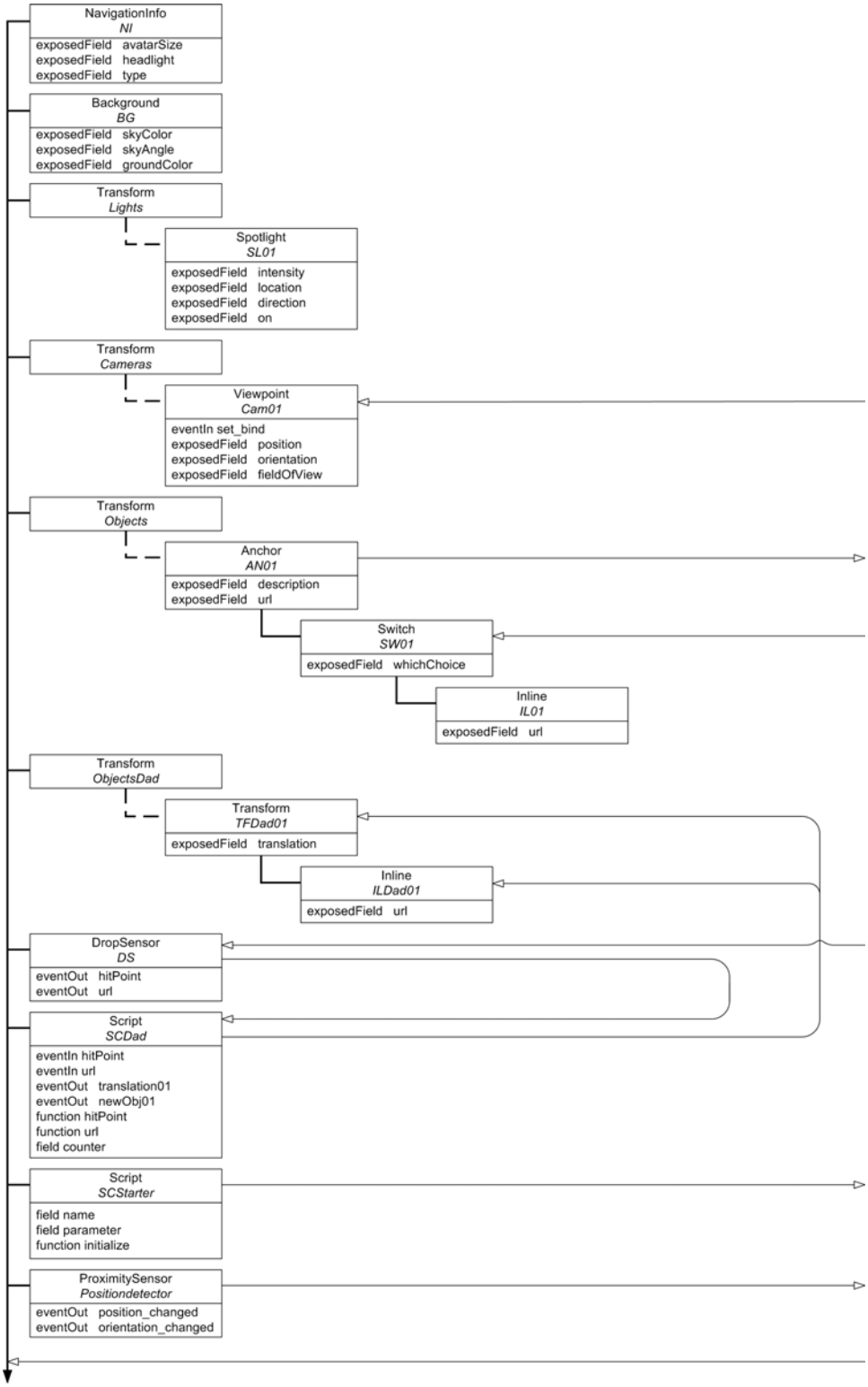

Abb. 46: VRML-Szenegraph der Datei root.wrz (eigener Entwurf und Darstellung)

Über die drei Felder des Background-Knotens ist definiert, wie der Hintergrund der Szene gestaltet ist. Der Hintergrund wird aus zwei unendlich großen Halbkugeln gebildet, die die virtuelle Welt von oben und unten umschließen. Die Felder skyColor und skyAngle steuern die Farbe bzw. einen Farbverlauf der oberen Halbkugel, die den Himmel darstellt. Hier wurden Werte ausgewählt, die einen Farbverlauf von hell- nach dunkelblau bewirken. Das Feld groundColor steuert die Farbe der unteren Halbkugel, die den Boden außerhalb der definierten Geometrie repräsentiert. Hier wurden Werte eingesetzt, die eine graue Farbe bewirken. Auf die Felder des Background-Elements kann ebenso wie auch auf andere Felder dynamisch zugriffen werden. So ist die Möglichkeit gegeben, das Aussehen des Himmels in Abhängigkeit von der lokalen Uhrzeit zu verändern, um einen Tag- bzw. alternativ einen Nachthimmel in der virtuellen Welt zu simulieren.

Der Transform-Knoten *Lights* ist ein Gruppenknoten, der die ihm untergeordneten Spotlight-Elemente zusammenfasst. Diese Spotlight-Elemente dienen dazu, die Szene gleichmäßig auszuleuchten. Über die Felder intensity, location und direction können Lichtintensität, Position der Lichtquelle und Abstrahlrichtung definiert werden. Durch die dynamische Veränderung der Werte dieser Felder besteht die Möglichkeit, die Beleuchtungsart an die lokale Tageszeit anzupassen oder spezifische Beleuchtungssituationen zu erzeugen. Auch diesen Punkt sieht das Konzept für die Funktionalität des Clients vor.

Der Transform-Knoten *Cameras* gruppiert die ihm untergeordneten Viewpoint-Elemente. Ein Viewpoint-Element repräsentiert eine Kamera in der virtuellen Welt. Über die Felder position, orientation und fieldOfView werden die Position der Kamera sowie deren Ausrichtung und Sichtfeld definiert. Das Feld set_bind erlaubt über die Zugriffsart eventIn das Auswählen bzw. Aktivieren einer bestimmten Kamera. Alle Felder der Viewpoint-Elemente werden über spezielle JavaScript-Programme angesteuert und ausgelesen bzw. dynamisch verändert (vgl. Kap. 6.5.1).

Der Transform-Knoten *Objects* ist für die Einbindung der gesamten Geometrie der Szene zuständig. Er gruppiert eine ganze Reihe von Anchor-Elementen, deren Anzahl der Zahl der einzelnen Gebäude im Neuenheimer Feld entspricht. Jedem dieser Anchor-Elemente sind wiederum jeweils ein Switch-Knoten und ein Inline-Knoten untergeordnet. Zum besseren Verständnis wird zunächst auf den hierarchisch niedrigsten Knoten eingegangen.

Die Inline-Elemente verweisen mit ihrem Feld url jeweils auf eine einzelne externe VRML-Datei, welche die eigentliche Geometrie enthält. Die Geometrie eines jeden

Gebäudes des Heidelberger Universitätscampus wird nach diesem Prinzip in die Szene geladen. Die zentrale VRML-Datei root.wrz enthält wie bereits angesprochen keine Geometrien, sondern nur Verweise, wo Geometrien liegen. So wird die in der Konzipierung des Systems angestrebte externe Referenzierung der Daten umgesetzt.
Über die den Inline-Elementen übergeordneten Switch-Knoten und die dynamische Ansteuerung deren whichChoice-Feldes wurde die Funktionalität umgesetzt, einzelne Geometrien nachzuladen bzw. ein- oder auszublenden (vgl. Kap.6.5).
Der Anchor-Knoten, der schließlich jedes Switch-Element sowie jeden Inline-Knoten und somit jede einzelne referenzierte Gebäudegeometrie umschließt, wird eingesetzt, um an zentraler Stelle die Vergabe von IDs und Hyperlinks zu bewerkstelligen. Dies geschieht über die Felder description und url. Damit wurde das Ziel erreicht, trotz der angestrebten externen Datenreferenzierung die Möglichkeit zu haben, jeder externen Referenz eine eigene ID in einem zentralen Dokument zuzuweisen. Trotzdem besteht daneben die Möglichkeit, diese IDs und Hyperlinks auch in den extern eingebundenen Dateien zu deklarieren. Hierbei wird diesen VRML-Dateien ein eigener Anchor-Knoten vorangestellt. In diesem Fall ist darauf zu achten, dass es nicht zu Überschneidungen kommt, die bewirken, dass eine über einen Inline-Knoten eingebundene Datei zwei IDs besitzt.
Die technische Besonderheit des Switch-Knotens, je nach Wert des whichChoice-Feldes einen bestimmten Kindknoten für die Anzeige auszuwählen, wird bewusst nicht genutzt, da es dann nicht möglich ist, die betreffenden Objekte zeitgleich einzublenden. Für Letzteres besteht dann Bedarf, wenn zwei Geometrien miteinander verglichen werden sollen. Dies ist beispielsweise bei den Varianten für die fünfte Neckarquerung der Fall.
Der Transform-Knoten ObjectsDad gruppiert mehrere Knotenpaare der Kombination Transform und Inline. Diese fangen diejenigen Objekte auf, die per Drag & Drop in das 3D-Fenster gezogen werden. Die Felder translation bzw. url dieser Knotenpaare werden für diese Funktionalität dynamisch mit Werten belegt (vgl. Kap. 6.5.5).
Die Knoten Dropsensor *DS* und Script *SCDad* sind ebenfalls Teil dieses Funktionsmoduls. Sie werden gesondert in Kapitel 6.5.6 beschrieben.
Der Script-Knoten *Starter* enthält ein Script. Es handelt sich dabei um ein VRMLScript, welches bei vollständigem Laden der VRML-Datei root.wrz ein externes JavaScript aufruft. Da dieses Element ebenfalls Teil eines komplexeren Funktionsmoduls ist, wird es in Kapitel 6.5.8 genauer beschrieben.

Der ProximitySensor-Knoten *Positiondetector* ist mit seinen Feldern der Zugriffsart eventOut dafür zuständig, die Bewegungen des Avatars in der Szene zu erfassen. Wie die entsprechenden Koordinatenwerte ausgelesen und weiterverarbeitet werden, wird in Kapitel 6.5.8 erläutert.

6.4.3 Cortona ActiveX Komponenten

Bei der Verwendung der ActiveX-Technologie unterscheidet man wie in Kapitel 6.1.7 beschrieben üblicherweise die drei Komponenten Control, Automation und Documents. Verwendung finden in dieser Arbeit nur die beiden erstgenannten. Eine ausführliche Beschreibung und Einführung in das Component Object Model (COM) und dessen Teilbereich ActiveX ist bei MICROSOFT (2007b) zu finden.
Die ActiveX Komponente Cortona Control ist für die Steuerung der Ansicht und die eigentliche Darstellung der VRML-Datei zuständig. Das Interface von Cortona Control stellt dazu eine Reihe von Eigenschaften, Methoden, Ereignismodellen und Objekten zur Verfügung (PARALLELGRAPHICS 2007b). Diese können je nach Bedarf mit spezifischen Parametern aufgerufen und mit Werten belegt werten. Cortona Control verändert dabei nicht den Inhalt des Szenegraphen.

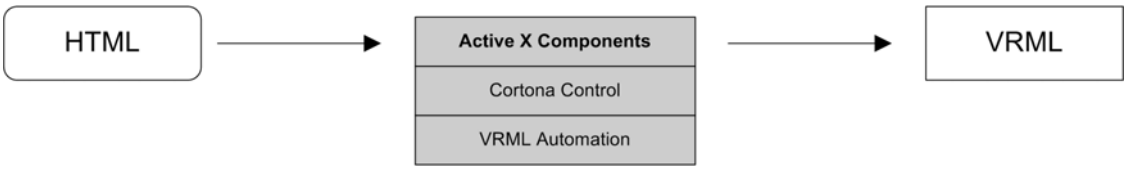

Abb. 47: Cortona ActiveX Komponenten als Schnittstelle zwischen HTML und VRML (eigene Darstellung)

Die ActiveX Komponente VRML Automation stellt verschiedene Schnittstellen bereit, mithilfe derer der Zugriff auf Inhalte des Szenegraphen ermöglicht wird. Die verschiedenen Interfaces von VRML Automation stellen ebenfalls bestimmte Eigenschaften und Methoden bereit, um die Objekte des VRML-Szenegraphen während der Laufzeit der Anwendung zu manipulieren (PARALLELGRAPHICS 2007c).
Abbildung 47 zeigt schematisch den Aufbau der Cortona ActiveX Komponenten, und wie das HTML-Dokument index.html über ActiveX auf das VRML-Dokument root.wrz zugreift. Welche Schnittstellen dieser beiden ActiveX Komponenten auf welche Weise genutzt werden, wird in Kapitel 6.5 näher ausgeführt.

6.5 Architektur der Funktionsmodule

Während Kapitel 6.4 einen Überblick über den Aufbau und die wesentlichen Elemente des HTML-Dokuments, des VRML-Dokuments und der Schnittstelle zwischen den beiden Dokumenten geboten hat, wird in diesem Kapitel die Funktionsweise jedes der spezifisch für diese Arbeit entwickelten Funktionsmodule erläutert. Wie im Konzept gefordert, ist es möglich, einzelne Module wegzulassen, ohne dass die Funktionalität der anderen Module dadurch eingeschränkt wird. Auf diese Art und Weise kann das Informationssystem funktional gezielt an verschiedene Ansprüche angepasst werden. Für vergleichsweise einfache Anwendungen wie reine Visualisierung und Abfrage von Basisinformationen kann das System mit wenigen Grundmodulen ausgestattet werden, während für anspruchsvollere Zwecke wie beispielsweise Darstellung von Bauvorhaben und Varianten sowie Abfrage von Objektinformationen aus Datenbanken entsprechend weitere Module dazu genommen werden. Die Programmierung der Funktionsmodule ist sehr komplex. Der Quellcode, der vom Verfasser dieser Arbeit dafür programmiert wurde, umfasst rund 3000 Zeilen. Um dennoch auch dem Laien einen Zugang zu erlauben, erfolgt die Beschreibung der Funktionsmodule nicht anhand des Quelltextes, sondern in einer schematischen und auf Diagrammen aufbauenden Art und Weise. Diese vereinfachen zwar, stellen aber im Gegenzug Aufbau und Funktionsweise für den Nichtfachmann deutlicher heraus. Die Schreibweise von HTML-Elementen bzw. VRML-Knoten orientiert sich dabei wie schon in den vorangegangenen Kapiteln an den Abbildungen, auf die Bezug genommen wird.

6.5.1 Basisinteraktion mit der Szene

Dieses Modul stellt wie in Kapitel 6.3 beschrieben eine Reihe verschiedenerer Basiswerkzeuge bereit, die notwendig sind, um mit der virtuellen 3D-Szene zu interagieren (Abb. 28 Bereich 2). Zum einen ist dies die Möglichkeit, aus einer Liste mit vorgegebenen Kameraansichten eine bestimmte auszuwählen bzw. zu aktivieren. Zum anderen kann über das Basismodul ausgewählt werden, wie man sich in der Szene fortbewegen will. Des Weiteren kann man die Darstellungsart der Geometrie in der Szene festlegen. Eine weitere Option ist es, für Innenräume einen speziellen an den Avatar gebundenen Scheinwerfer zu aktivieren.
Abbildung 48 zeigt den Ausschnitt des DOM-Baums, welcher für die Funktionalität dieses Moduls relevant ist. Die relative Anordnung der Elemente innerhalb der Baumhierarchie entspricht der in Abbildung 44. Die grau unterlegten, abgerundeten Felder

symbolisieren JavaScript-Programme. Die beschrifteten Pfeile zwischen einzelnen Elementen der Abbildung stellen Aufrufe von Programmen bzw. Eigenschaften oder Methoden dar. In Klammern stehen dabei jeweils Parameter bzw. Werte.

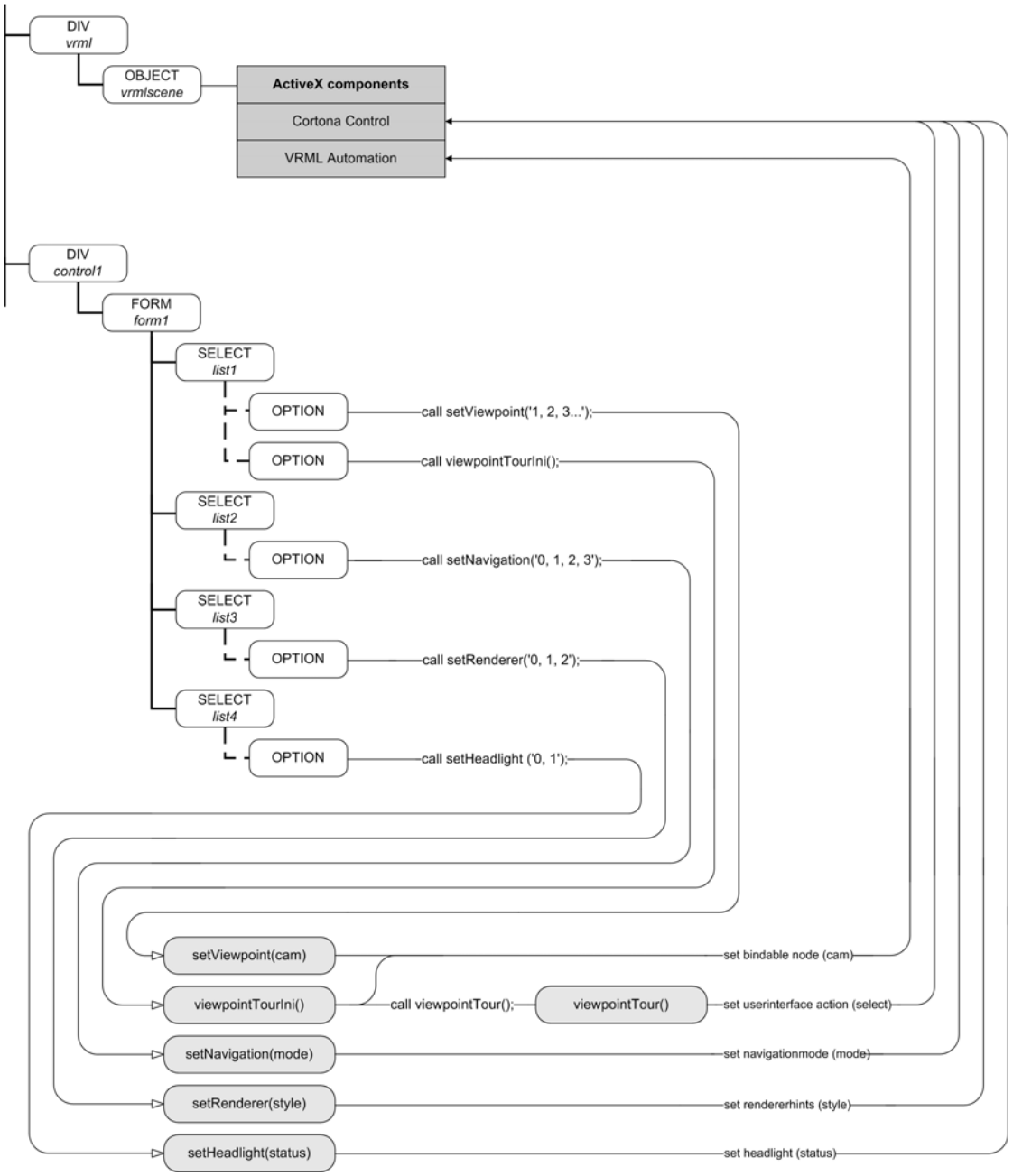

Abb. 48: Modul für die Basisinteraktion mit der Szene (eigener Entwurf und Darstellung)

Relevant ist zunächst das DIV-Element *control1* mit seinen ihm hierarchisch nachgeordneten Knoten. Diese nehmen die Benutzerwünsche zur Interaktion mit der Szene auf und verarbeiten diese in bestimmten JavaScript-Programmen. Diese Programme wiederum rufen entsprechende Eigenschaften und Methoden in den ActiveX Komponenten auf, die über das OBJECT-Element *vrmlscene* des DIV-Knotens *vrml* bereit stehen.
Über die Auswahl eines OPTION-Eintrages im SELECT-Knoten *list1* wählt der Benutzer eine Kamera innerhalb der dreidimensionalen Szene aus. Bei der Wahl eines Listeneintrags wird der Absendewert des übergeordneten Formular-Elements *form1* bestimmt. Die Auswahl aktiviert zunächst einen Event-Handler des Typs onChange (vgl. Kap. 6.1.4). Statt eines Wertes wird nun das JavaScript-Programm setViewPoint(cam)aufgerufen. Dieses Programm erwartet für seinen Parameter cam einen Wert. Ein solcher ist an jeden Listeneintrag gebunden und wird gemeinsam mit dem Aufruf des JavaScripts übergeben. Bei dem Wert handelt es sich um eine Zahl zwischen 1 und 50, was der Anzahl der in der VRML-Szene definierten Kamera-Knoten entspricht. Das Programm setviewPoint(cam) gleicht intern den übergebenen Wert mit einer Referenzliste ab, welche die Namen enthält, die die Kamera-Knoten in der VRML-Datei root.wrz tragen. Nun greift das JavaScript auf das VRMLNode-Interface der ActiveX Komponente VRML Automation zu, wählt den Viewpoint-Knoten mit dem dem Parameter cam entsprechenden Namen aus und belegt dessen Feld set_bind mit einem Wert, der die Kamera aktiviert (Abb. 6.2.4).
Neben denjenigen Listeneinträgen, die es erlauben, auf einzelne statische Kameras oder animierte Kamerafahrten zuzugreifen (Abb. 31), enthält der SELECT-Knoten list1 den Eintrag „geführte Tour“. Dieser Listeneintrag ruft das JavaScript-Programm viewpointTourIni() auf. Dieses Programm erwartet keine weiteren Parameter; eine Wertübergabe an das Programm findet nicht statt. Das Script greift als erstes auf die ActiveX Komponenten VRML Automation zu und aktiviert denjenigen Viewpoint, der bei jedem Start des Systems als Standardansicht ausgewählt ist. Dies geschieht analog zum Funktionsablauf des Scriptes setViewpoint(cam). Danach ruft das Script das JavaScript-Programm viewpointTour() auf. Dieses greift auf die Methode userinterface action der ActiveX Komponente Cortona Control zu. Mittels dieser Methode und dem an sie übergebenen Wert wird der jeweils nächste Viewpoint im VRML-Szenegraphen aktiviert werden. Das Programm viewpointTour() ruft sich in voreingestellten Zeitintervallen selbst auf und aktiviert über die beschriebene Methode den jeweils nächsten Viewpoint. Dies geschieht so lange, bis der Benutzer einen anderen Kamerastandpunkt

aus der Liste auswählt. In diesem Fall wird wie beschrieben das JavaScript-Programm setViewpont(cam) aufgerufen. Dieses hält neben der bereits beschriebenen Funktionalität eine Routine vor, die das Intervall des Scripts viewpointTour() unterbricht.
Über die Auswahl eines Eintrages im SELECT-Knoten *list2* bestimmt der Benutzer die Navigationsart des Avatars. Durch die Auswahl eines Listeneintrags wird das JavaScript-Programm setNavigation(mode) aufgerufen. Dieses erwartet für seinen Parameter mode einen Wert, welcher beim Aufruf der Funktion übergeben wird. Der Wertebereich für diesen Parameter liegt zwischen 0 und 4. Das Script greift dann auf die Eigenschaft navigationmode der ActiveX Komponente Cortona Control zu. Dort werden dann die Werte dieser Eigenschaft in Abhängigkeit des Parameters mode verändert und so der gewünschte Navigationsmodus aktiviert.
Der SELECT-Knoten *list3* erlaubt dem Nutzer die Beeinflussung der Darstellungsart der Szene. Wählt er einen Listeneintrag aus, wird das Programm setRenderer(style) aufgerufen. Beim Aufruf dieser Funktion werden in Abhängigkeit des jeweiligen Listeneintrags bestimmte Werte an den Parameter style übergeben. Die Funktion wertet die übergebenen Werte aus und greift auf die Eigenschaft rendererhints der ActiveX Komponente Cortona Control zu. Die Werte dieser Eigenschaft werden entsprechend dem Parameter style verändert. So wird die vom Benutzer verlangte Art der Darstellung aktiv.
Mit der Auswahl eines Eintrages im SELECT-Element *list4* steuert der Benutzer den Betriebszustand der an den Avatar gebundenen Lichtquelle. Bei Auswahl eines Listeneintrags wird die Funktion setHeadlight(status) aufgerufen. Deren Parameter status wertet den dem Listeneintrag zugeordneten Wert aus. Danach wird die Eigenschaft headlight von Cortona Control aufgerufen und deren Wert analog zum Parameter status verändert. Auf diese Weise wird die Lichtquelle an- bzw. ausgeschaltet.

6.5.2 Erweiterte Steuerung der Szenenansicht

Dieses Modul stellt wie in Kapitel 6.3.3 beschrieben zusätzlich zu den Basiswerkzeugen eine Reihe von Möglichkeiten bereit, die Ansicht der Szene bzw. die Art der Navigation zu beeinflussen (Abb. 28 Punkt 3). Zum einen kann der Benutzer die zuletzt ausgewählte Viewpoint-Ansicht wiederherstellen. Zum anderen besteht die Möglichkeit, das Sichtfeld des aktuell aktiven Viewpoints zu verändern. Darüber hinaus kann zusätzlich zu der bereits aktivierten Hauptnavigationsart ein spezieller Submodus bzw. eine erweitere Navigationsart ausgewählt werden.

Abbildung 49 zeigt die Abschnitte des DOM-Baums, welche für die Funktionalität dieses Moduls relevant sind. Die relative Anordnung der Elemente innerhalb der Baumhierarchie entspricht der in Abbildung 44. Die grau unterlegten, abgerundeten Felder symbolisieren JavaScript-Programme. Die beschrifteten Pfeile zwischen einzelnen Elementen der Abbildung stellen Aufrufe von Programmen bzw. Eigenschaften oder Methoden dar. In Klammern stehen dabei jeweils Parameter bzw. Werte.

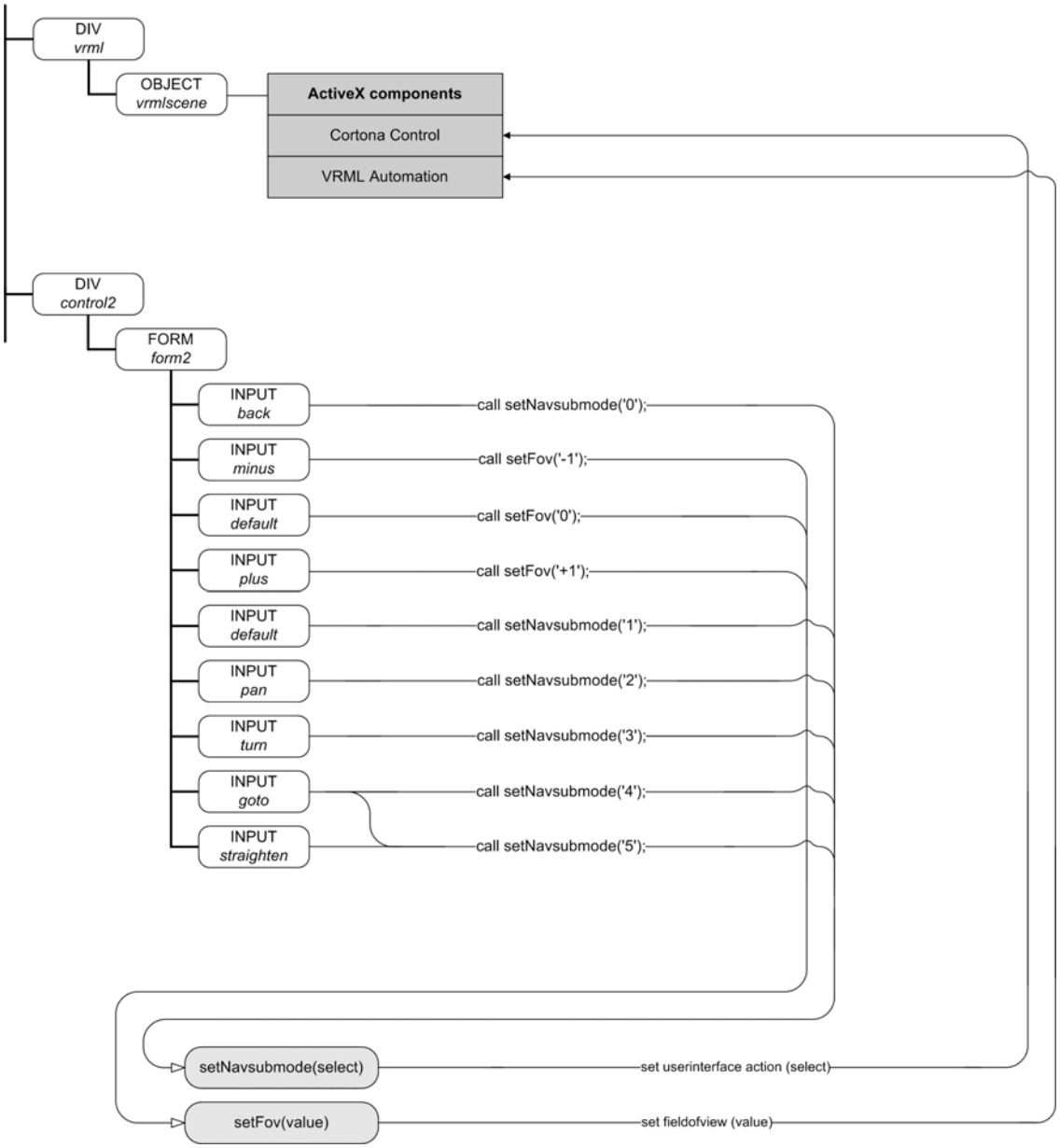

Abb. 49: Modul für die erweiterte Steuerung der Szenenansicht (eigener Entwurf und Darstellung)

Relevant ist zuerst das DIV-Element *control2* mit seinen hierarchisch untergeordneten Knoten. Diese nehmen die Benutzereingaben zur Interaktion auf und verarbeiten sie in speziell dafür programmierten JavaScript-Funktionen. Diese Funktionen rufen entsprechende Eigenschaften und Methoden in den ActiveX Komponenten auf, die über das OBJECT-Element *vrmlscene* des DIV-Knotens *vrml* bereit stehen.
Über die Auswahl eines INPUT-Elements im Formular-Knoten *form2* des DIV-Elementes *control2* wird ein Event-Handler onClick aktiv, der den Aufruf des für die Funktion notwendigen JavaScript-Programms unter Übergabe der entsprechenden Parameterwerte steuert.
Genau genommen handelt es sich bei diesen Elementen nicht um INPUT-Knoten, sondern um DIV-Elemente, denen das Verhalten von INPUT-Elementen gegeben wurde. Dieser Schritt wurde durchgeführt, um mehr Einfluss auf das Aussehen der graphischen Benutzeroberfläche zu haben. Mit Rücksicht auf den schematischen Charakter der Abbildung und der Beschreibung wird hier die Typbezeichnung INPUT verwendet.
Über das INPUT-Element *back*, welches hinter der Schaltfläche „zurück“ in Abbildung 27 steht, wird die Funktion setNavsubmode(select) aufgerufen. Der Parameter select wird dabei mit dem dem INPUT-Element zugeordneten Wert belegt. Die Funktion greift danach auf die Methode userinterface action der ActiveX Komponente Cortona Control zu. In Abhängigkeit des an diese Methode per select übermittelten Wertes wird das Ansichtfenster bzw. die Position des Avatars auf den zuletzt ausgewählten Viewpoint zurückgesetzt.
Die INPUT-Elemente *minus*, *default* und *plus*, die sich hinter den Schaltflächen „-“, „45°“ und „+“ in Abbildung 27 verbergen, rufen bei einem onClick-Event mit jeweils unterschiedlichen Werten das JavaScript-Programm setFov(value) auf. Diese Funktion deklariert unter anderem eine dokumentweit gültige globale Variable namens fov, die den jeweils aktuellen Wert für das Sichtfeld der aktiven Ansicht darstellt. Die Maßzahl wird dabei als Bogenmaß bzw. Radiant (rad) angegeben und bewegt sich in einem Wertebereich von 0 bis 2pi. Die Funktion wertet die an den Parameter value übergebenen Werte aus und greift auf die Methode fieldofview von Cortona Control zu. Mittels dieser Methode kann das Sichtfeld der aktiven Ansicht verändert werden.
Der durch das INPUT-Element *minus* an den Parameter value übergebene Wert '-1' wird dabei einer Variablen fovminus mit dem Wert -0,09 rad zugeordnet. Dies entspricht in etwa einem Gradmaß von -5°. Analog ist der Vorgang beim INPUT-Element *plus*. Nun verrechnet das JavaScript-Programm die Variablen fov und fovminus bzw. fovplus miteinander und übergibt den Ergebniswert an die Methode fieldofview. Da der

aktuelle Wert immer in der Variablen fov gespeichert ist, wird bei mehrfachem Aufruf mit positiven bzw. negativen Werten das Sichtfeld schrittweise um 5° verkleinert bzw. vergrößert.
Das INPUT-Element *default* ruft die Funktion setFov(value) mit einem Wert von '0' auf. Hierdurch wird der Variablen fov ein Wert von 0,79 rad zugewiesen, was in etwa einem Gradmaß von 45° entspricht. Dieser Wert wird der Methode fieldofview übergeben. So wird das Sichtfeld der aktuellen Ansicht auf den Standardwert zurückgesetzt, der der menschlichen Wahrnehmung entspricht.
Die INPUT-Elemente *default*, *pan* und *turn* entsprechen den Schaltflächen „def“, „versch“ und „dreh“ in Abbildung 27. Die Funktionsweise dieser Elemente gleicht denen des INPUT-Knotens *back*. Je nach Wertübergabe für den Parameter der Funktion setNavsubmode(select) wird die Methode userinterface action der ActiveX Komponente Cortona Control mit spezifischen Parametern angesteuert.
Hinter Schaltfläche „ziel“ in Abbildung 27 verbirgt sich das INPUT-Element *goto*. Bei einem onClick Event wird ebenfalls das JavaScript-Programm setNavsubmode(select) aufgerufen, um dann in der beschriebenen Weise die Bereitstellung der in Kapitel 6.3.3 dargestellten Funktionalität zu bewirken. In diesem Fall ruft sich die Funktion setNavsubmode(select) nach einem kurzen Zeitintervall nochmals selbst auf; dieses Mal jedoch unter Übergabe desjenigen Wertes für den Parameter select, der auch über die zum INPUT-Element *straighten* gehörige Schaltfläche „horiz“ übergeben wird. Der Nachteil der in Kapitel 6.3 beschriebenen Funktionalität „ziel“, den Viewpoint senkrecht an der angeklickten Oberfläche auszurichten, wird dadurch gemindert.
Das letzte INPUT-Element des Formulars *form2* entspricht der Schaltfläche „horiz“ in Abbildung 27. Seine Funktionsweise läuft analog der bereits beschriebenen Aufrufe des JavaScript-Programms setNavsubmode(select) ab. Die Methode userinterface action von Cortona Control wird dabei mit einem Parameter aufgerufen, der das horizontale Ausrichten der aktiven Benutzeransicht bewirkt.

6.5.3 Absolute Positionierung des Avatars

Dieses Modul stellt die in Kapitel 6.3.5 beschriebene Funktionalität zur absoluten Positionierung des Avatars mittels Klicken auf die Übersichtskarte bereit. Abbildung 50 zeigt die Abschnitte des DOM-Baums, welche für die Funktionalität dieses Moduls relevant sind. Die relative Anordnung der Elemente innerhalb der Baumhierarchie entspricht der in Abbildung 44. Die grau unterlegten, abgerundeten Felder symbolisieren

JavaScript-Programme. Die beschrifteten Pfeile zwischen einzelnen Elementen der Abbildung stellen Aufrufe von Programmen bzw. Eigenschaften oder Methoden dar. In Klammern stehen dabei jeweils Parameter bzw. Werte, welche erwartet bzw. übergeben werden.

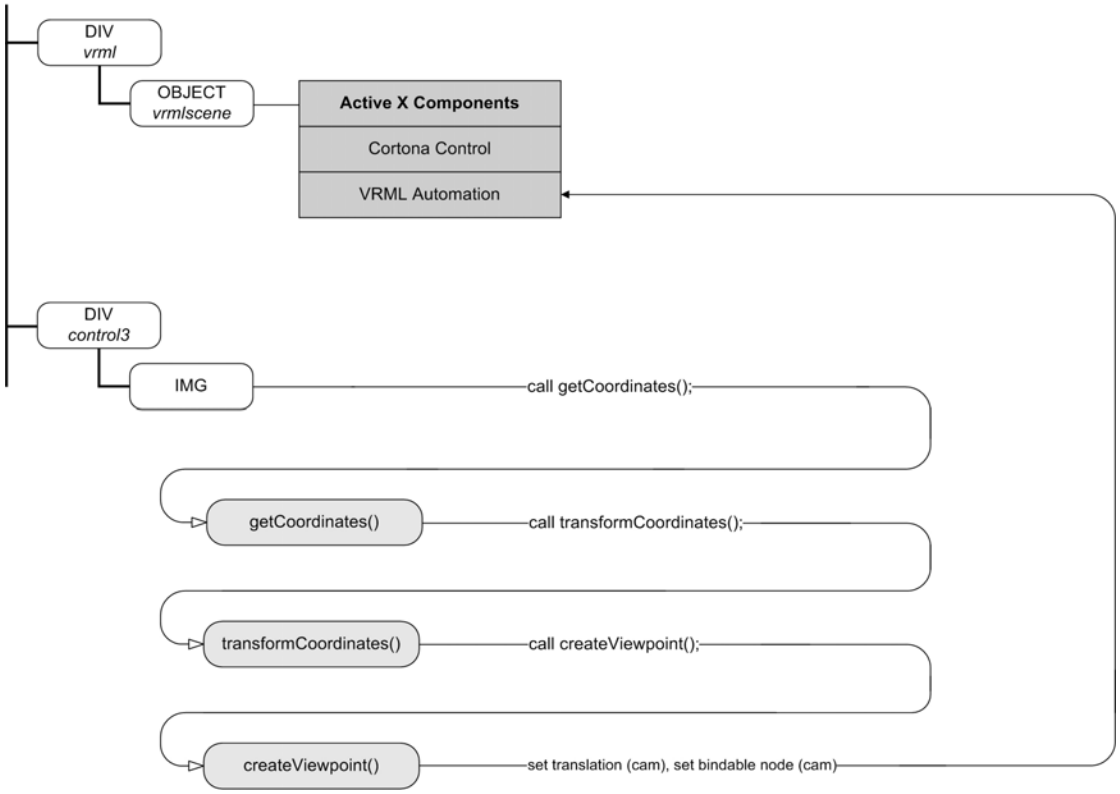

Abb. 50: Modul für die absolute Positionierung des Avatars (eigener Entwurf und Darstellung)

Relevant ist zunächst das DIV-Element *control3* mit seinem hierarchisch untergeordneten IMG-Knoten. Dieser nimmt die Benutzereingaben zur Interaktion auf und leitet sie zur Verarbeitung an spezifische JavaScript-Funktionen. Diese Funktionen rufen entsprechende Eigenschaften und Methoden in den ActiveX Komponenten auf, die über das OBJECT-Element *vrmlscene* des DIV-Knotens *vrml* bereit stehen.
Das IMG-Element des DIV-Knotens *control3* liegt deckungsgleich und für den Benutzer nicht sichtbar über dem IMG-Element des DIV-Knotens *map*. Es ist mit einem onClick Event-Handler verknüpft, der bei einem Mausklick des Benutzers auf den Bereich der Karte das JavaScript-Programm getCoordinates() aufruft. Der Aufruf dieser Funktion

erfolgt ohne die Übergabe von Werten. Das JavaScript getCoordinates() deklariert die dokumentweit gültigen globalen Variablen mapx und mapy. Dann nutzt das Script die Eigenschaften offsetX und offsetY des Event-Objekts des Browsers, um die objektrelative Mauszeiger-Position zu erfassen. Diese Position bzw. deren Koordinaten x und y werden in den Variablen mapx und mapy gespeichert. Dabei handelt es sich um diejenige Position, die der Mauszeiger zum Zeitpunkt des onClick-Ereignisses relativ zur linken oberen Ecke des IMG-Elementes eingenommen hat. Auf diese Art und Weise werden in einem ersten Schritt relative Koordinaten gespeichert, die in einem definierbaren Verhältnis zu der in der Benutzeroberfläche abgebildeten Übersichtskarte stehen. Das JavaScript getCoordinates() ruft nach Speicherung der relativen Koordinaten das Programm transformCoordinates() auf. Dieses ist dafür zuständig, die Koordinaten der relativen Maßstabsebene der Karte in absolute Koordinatenwerte umzurechnen, die für die Positionierung des Avatars in der VRML-Szene notwendig sind. Diese Umrechnung erfolgt auf Grundlage einer affinen Koordinatentransformation (BILL 1999a, 137 ff.). Die transformierten Koordinatenwerte werden in den dafür deklarierten absoluten Variablen mapxtr und mapytr abgespeichert, die einen dokumentweiten Zugriff ermöglichen. Nach Abschluss der Transformation und Speicherung des transformierten Koordinatenpaares ruft die Funktion transformCoordinates() das JavaScript-Programm createViewpoint() auf. Diese Funktion erwartet keine Parameter, sondern greift auf die globalen Variablen mit den absoluten Koordinatenwerten zu. Um diese Werte an einen dafür vorgesehenen benutzerdefinierbaren Viewpoint in der VRML-Szene zu übergeben, greift die Funktion auf das VRMLNode-Interface der ActiveX Komponente VRML Automation zu. Über dieses wird der benutzerdefinierbare Viewpoint ausgewählt und in einem ersten Schritt dessen Feld position mit den transformierten Koordinaten belegt (Abb. 46). Als Wert für die Höhe wird an dieser Stelle ein Standardwert von 1,6m übergeben. In einem weiteren Schritt belegt das Skript das Feld set_bind über die Zugriffsart eventIn mit einem Wert, der die Aktivierung des benutzerdefinierbaren Viewpoint-Knotens bewirkt (Abb. 46).

Da die Positions-Werte dieses Knotens in Variablen auf HTML-Seite abgespeichert bleiben, bleibt der Standpunkt des benutzerdefinierten Viewpoints solange erhalten und für den Nutzer aufrufbar, bis er über ein onClick-Ereignis auf die Übersichtskarte mit neuen Koordinaten belegt wird.

6.5.4 Steuerung der Szeneninhalte

Dieses Modul stellt die in Kapitel 6.3.4 beschriebene Funktionalität zur Bestimmung der Inhalte der Szene bereit. Abbildung 51 zeigt die Abschnitte des DOM-Baums, welche für die Funktionalität dieses Moduls relevant sind. Die relative Anordnung der Elemente innerhalb der Baumhierarchie entspricht der in Abbildung 44. Die grau unterlegten, abgerundeten Felder symbolisieren JavaScript-Programme. Die beschrifteten Pfeile zwischen einzelnen Elementen der Abbildung stellen Aufrufe von Programmen bzw. von Eigenschaften oder von Methoden dar. In Klammern stehen dabei jeweils Parameter bzw. Werte, welche erwartet bzw. übergeben werden.
Für die Eingaben des Benutzers ist das DIV-Element *control4* mit seinen hierarchisch untergeordneten Formular-Knoten (FORM) zuständig. Diese nehmen die Eingaben zur Interaktion auf und leiten sie zur Verarbeitung an spezifische JavaScript-Funktionen weiter. Diese Funktionen rufen entsprechende Eigenschaften und Methoden in den ActiveX Komponenten auf, die über das OBJECT-Element *vrmlscene* des DIV-Knotens *vrml* bereit stehen. Da sich Funktionalität und technischer Aufbau, die sich hinter den FORM-Knoten *form3* und *form4* bzw. *form5* und *form6* verbergen, erheblich voneinander unterscheiden, ist die Beschreibung auf zwei Unterkapitel verteilt. In diesem werden die ersten beiden beschrieben. Dem Aufbau und der Programmierung der letzten beiden ist das anschließende Kapitel gewidmet.
Technischer Aufbau und Implementierung der FORM-Knoten *form3* und *form4* gleichen sich im Wesentlichen. Nachfolgend wird deswegen nur auf die Funktionsweise des FORM-Knotens *form3* eingegangen. Die rund dreißig INPUT-Elemente dieser FORM-Knoten sind wie schon erwähnt als Checkboxen definiert, deren Auswahlzustand durch ein Kreuz angezeigt wird. Bei Aktivierung bzw. Deaktivierung einer solchen Box wird ein onClick Event-Handler aktiv. Der Event-Handler ruft das JavaScript-Programm setSwitch(select) auf. Dies steuert die Werte der Felder aller SWITCH-Knoten im VRML-Szenegraphen. Der aktuelle Wert ist nicht nur im entsprechenden Feld whichChoice der SWITCH-Knoten in der VRML-Szene abgelegt (Abb. 46). Auch im HTML-Dokument bzw. in der JavaScript-Funktion setSwitch(select) sind globale Variablen definiert, in denen der Zustand sowohl jedes einzelnen SWITCH-Knotens wie auch jeder einzelnen Checkbox abgespeichert ist. Über den Wert, der in Abhängigkeit von jeder einzelnen Checkbox an den Parameter select der Funktion übergeben wird, ist definiert, auf welchen SWITCH-Knoten zugegriffen werden muss. Das JavaScript ruft nun die diesem SWITCH-Knoten im HTML-Dokument zugeordnete

Variable auf und verändert deren Wert. Zeitgleich greift die Funktion auf das VRMLNode-Interface der ActiveX Komponente VRML Automation zu, wählt den entsprechenden SWITCH-Knoten und das dazugehörige Feld whichChoice aus, um dessen Wert zu ändern (Abb. 46). Je nachdem, ob dieser Wert -1 oder 0 beträgt, wird einer der dem SWITCH-Knoten untergeordneten INLINE-Knoten aktiviert oder deaktiviert.

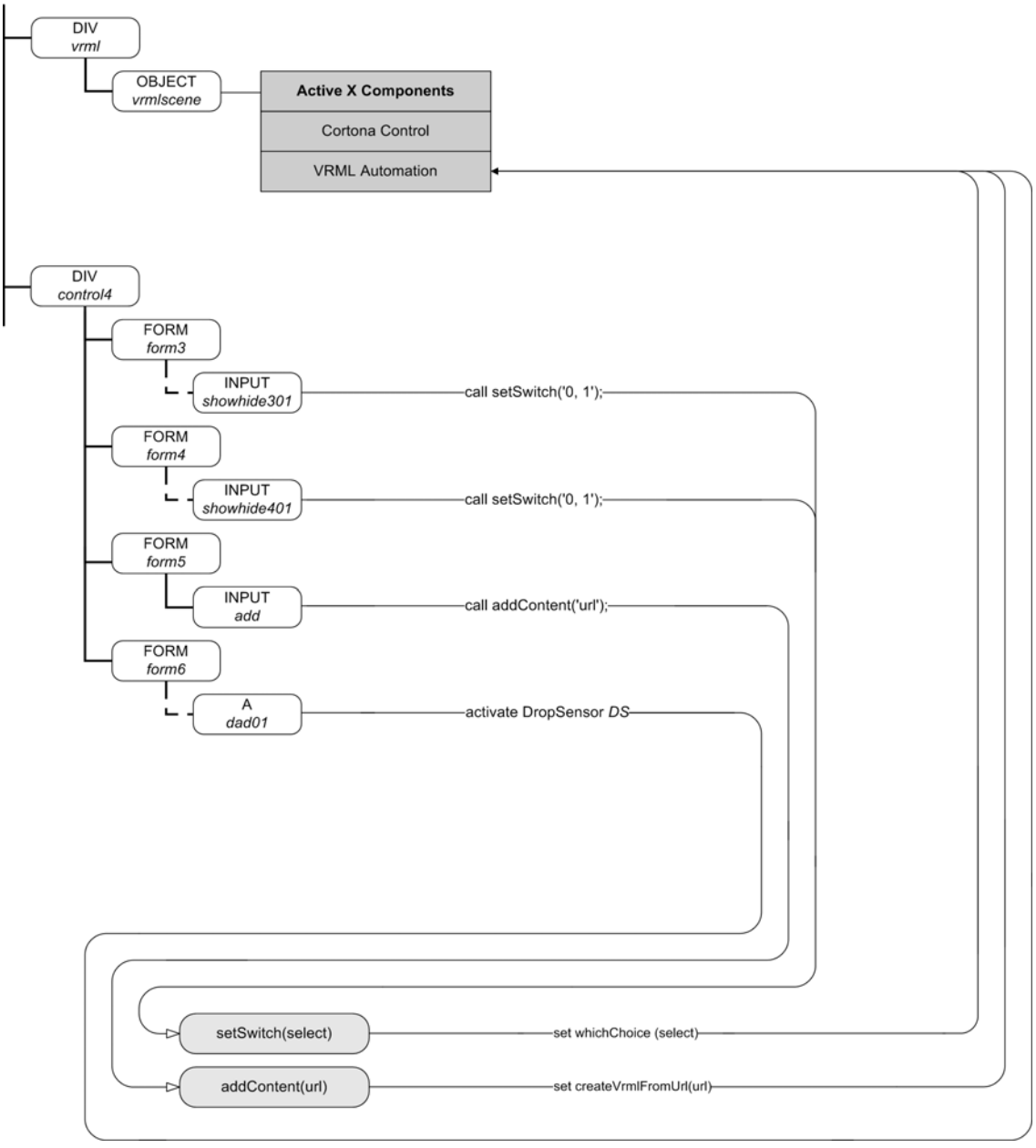

Abb. 51: Modul zur Steuerung von Szeneninhalten (eigener Entwurf und Darstellung)

6.5.5 Benutzerdefinierte Erweiterung von Szeneninhalten

Dieses Modul stellt die in Kapitel 6.3.4 beschriebene Funktionalität zur Erweiterung der Szeneninhalte durch den Benutzer dar. Abbildung 51 stellt den Funktionsablauf dieses Moduls dar. Hinsichtlich der Abbildung gelten für die nachfolgende Beschreibung dieselben Vorbemerkungen wie im vorangegangenen Kapitel.
Über das INPUT-Element des FORM-Knotens *form5* wird dem Benutzer ein einzeiliges Eingabefeld sowie eine Schaltfläche zur Verfügung gestellt (Abb. 40). Diese beiden Werkzeuge ermöglichen es dem Nutzer, der Szene weitere Inhalte hinzuzufügen. Das System ist dabei in der Lage, sowohl auf lokale Inhalte als auch auf externe Inhalte zuzugreifen.
Über das Eingabefeld wird der Uniform Ressource Locator derjenigen Datei bekannt gemacht, die der Szene als neuer Inhalt hinzugefügt werden soll. Dies geschieht durch eine entsprechende Pfadangabe sowie den Dateinamen inklusive Dateiendung. Die Schaltfläche „Laden“ aktiviert ein onClick-Event, welches das JavaScript-Programm addContent(url) aufruft. Als Parameter url erwartet diese Funktion den im Eingabefeld notierten URL. Das JavaScript ruft nun die Methode createVRMLFromUrl der ActiveX-Komponente Cortona Control auf und übergibt dieser den URL. Die VRML-Datei, auf die der URL zeigt, wird daraufhin intern geladen. Sobald dies ausgeführt ist, ruft das Programm addContent(url) in Cortona Control die Eigenschaft RootNodes mit der Anweisung Add auf. Dadurch werden die intern neu geladenen VRML-Inhalte dem aktuellen VRML-Szenegraphen hinzugefügt. Dieser Schritt ist in Abbildung 46 durch den direkt auf die oberste Ebene des Szenegraphen zeigenden Pfeil schematisch dargestellt. Die Darstellung der auf diese Art und Weise vom Benutzer hinzugefügten Inhalte wird dabei in Echtzeit übernommen, ohne dass ein Neuladen der Gesamtszene notwendig wird.
Über die im Dokument im FORM-Knoten *form6* implementierten Anchor-Elemente (A) bzw. den damit verknüpften Hyperreferenzen (vgl. Kap. 6.1.2) kann der Nutzer aus einer vorgegebenen Menge an Musterobjekten diejenigen auswählen, die er der aktuellen Szene als neue Inhalte hinzufügen möchte (Abb. 41). Diese Auswahl geschieht wie in Kapitel 6.3.4 beschrieben über eine Drag & Drop Methode.
Dabei wird direkt und zunächst ohne Aufruf einer im HTML-Dokument referenzierten JavaScript-Funktion über die ActiveX-Komponente Cortona Control auf den VRML-Szenegraphen zugegriffen. Dieser Vorgang wird durch den in Abbildung 51 vom Anchor-Element dad01 ausgehenden Pfeil und den in Abbildung 46 auf den

DropSensor-Knoten *DS* zeigenden Pfeil veranschaulicht. Die eigentliche Funktionalität dieses Moduls ist im Gegensatz der meisten anderen Funktionen zum größten Teil im Szenegraphen des VRML-Dokuments root.wrz implementiert. Dies wird anhand von Abbildung 52 deutlich. Sie zeigt die relevanten Teile des Szenegraphen. Die relative Anordnung der Knoten entspricht dabei der in Abbildung 46.
Die vom Benutzer per Drag & Drop in das 3D-Fenster gezogene Hyperreferenz liefert dem DropSensor zwei verschiedene Informationen. Dies ist zum einen der mit der Hyperreferenz verbundene Uniform Ressource Locator, der den Ort und den Namen der Datei enthält, die als neuer Szeneninhalt hinzugefügt werden soll. Zum anderen definiert das drop-Ereignis die Koordinaten in der 3D-Szene, an der der neue Szeneninhalt positioniert werden soll. Damit keines der über diese Methode hinzugefügten Objekte frei im dreidimensionalen Raum schwebt, werden diese Raumkoordinaten auf eine zweidimensionale Ebene projiziert. Diese beiden Informationen verarbeitet der DropSensor und stellt sie in den beiden eventOut Feldern hitPoint bzw. url für den weiteren Zugriff bereit (Abb. 52). Sobald sich die Werte in diesen Feldern ändern, was bei jedem vom Nutzer ausgeführten Drag & Drop Ereignis der Fall ist, wird eine Kommunikation mit dem Script-Knoten *SCDad* aufgebaut. Diese Kommunikation findet in einer für jedes Feld gesonderten VRML-Route (ROUTE) statt (vgl. Kap. 6.1.6). Der Script-Knoten *SCDad* ruft bei jeder derartigen Kommunikation zwei JavaScript-Programme auf. Die Funktion hitpoint legt dabei die bei jedem Drag & Drop Ereignis erzeugten Koordinaten in einer Variablen ab. Die Funktion url weist den bei jedem Drag & Drop Ereignis übergebenen URL ebenfalls einer Variablen zu. Die beiden Funktionen stellen diese Variablen über die eventOut-Felder translation bzw. newObj des Script-Knotens für den weiteren Zugriff bereit (Abb. 52). Der Wert des Feldes translation, welcher die Koordinaten repräsentiert, wird dann über eine VRML-Route (ROUTE) an einen noch nicht belegten Transform-Knoten *TFDad* bzw. dessen Feld translation übergeben. Damit wird die Position der diesem Transform-Knoten untergeordneten Knoten definiert. Zeitgleich wird der Wert des Feldes newObj vom Script-Knoten an den Inline-Knoten *ILDad* desselben Transform-Knotens übergeben. Auch diese Kommunikation wir durch eine VRML-Route realisiert. Damit wird diejenige Datei sowie deren Speicherort definiert, die geladen werden soll. Das über die Drag & Drop Methode als neuer Szeneninhalt hinzugefügte Objekt wird ebenfalls in Echtzeit übernommen, ohne dass die Szene neu geladen werden muss.

Im Vergleich zur Definition neuer Inhalte über das INPUT-Element des Formular-Knotens *form5* werden bei der Drag & Drop Methode dem VRML-Szenegraphen keine neuen Knoten hinzugefügt, sondern Werte bereits existierender Knoten verändert.
Eine Speicherung der auf eine der beiden Arten vom Nutzer aktiv eingebrachten Szeneninhalte kann mithilfe vorgefertigter Module aus dem Software Developement Kit der Cortona ActiveX Komponente realisiert werden (PARALLELGRAPHICS 2007b). Dies gilt auch für die Speicherung der aktuellen Szenenansicht als einzelnes Bild.

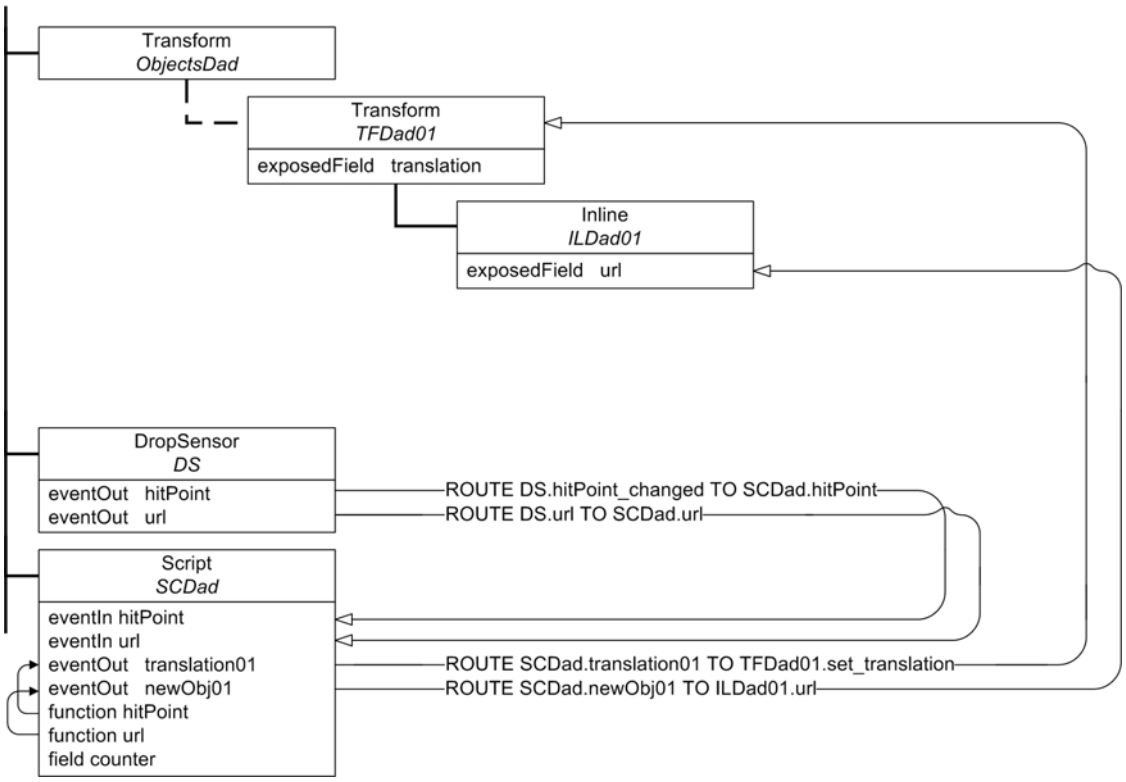

Abb. 52: Modul für die Erweiterung von Szeneninhalten im VRML-Dokument (eigener Entwurf und Darstellung)

6.5.6 Interaktive Neupositionierung von Szeneninhalten

Dieses Modul stellt die in Kapitel 6.3.4 beschriebene Funktionalität zum Verschieben der über die Drag & Drop Methode neu generierten Szeneninhalte dar.
Wie auch schon der im vorangegangenen Kapitel beschriebene Funktionsablauf des Moduls zur Erweiterung der Szeneninhalte durch den Benutzer über Drag & Drop ist

auch dieses Modul im VRML-Szenegraphen implementiert. Dies kann entweder im Szenegraphen des VRML-Hauptdokuments root.wrz oder aber im Szenegraphen derjenigen VRML-Datei realisiert sein, welche die eigentlichen Informationen zur Geometrie eines Objektes enthält. Nachfolgend wird die Funktionsweise an einer VRML-Datei erläutert, die dem Benutzer über eine Hyperreferenz des Formular-Elements *form6* zur Verfügung gestellt wird. Abbildung 53 stellt den Funktionsablauf dieses Moduls dar. Hinsichtlich der Abbildung gelten für die nachfolgende Beschreibung dieselben Vorbemerkungen wie in den vorangegangenen Kapiteln. Zudem zeigt die Abbildung auch nur die Teile des VRML-Szenegraphen, die für die Ausführung der Funktionalität notwendig sind.

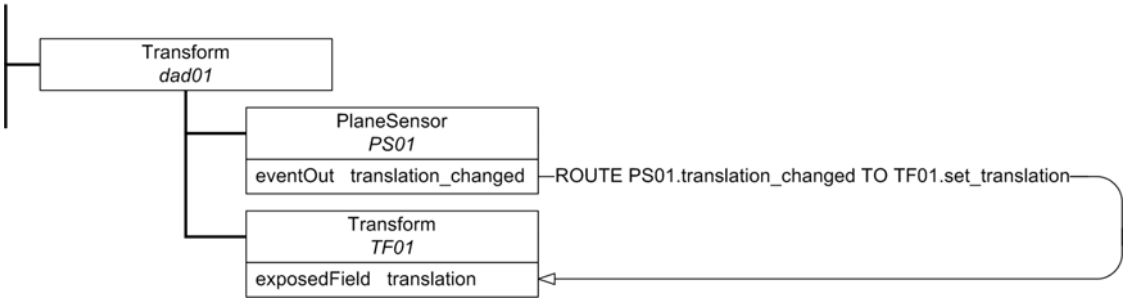

Abb. 53: Modul für interaktives Neupositionieren von Objekten (eigener Entwurf und Darstellung)

Wesentliche Elemente dieses Moduls sind der PlaneSensor *PS01* sowie die dazugehörige VRML-Route (ROUTE). Sobald der Benutzer mit dem Mauszeiger ein Objekt im 3D-Fenster berührt, welches mit einem solchen PlaneSensor ausgestattet ist, wird dieser aktiv. Führt der Benutzer nun mit der Maus eine Ziehbewegung aus, so liefert der PlaneSensor in seinem eventOut Feld translation_changed die auf eine zweidimensionale Ebene projizierten Positionswerte des Mauszeigers in der 3D-Szene. Sobald sich die Werte in dem Feld translation_changed ändern, wird eine VRML-Route (ROUTE) aufgerufen, die die Kommunikation mit dem Transform-Knoten *TF01* aufnimmt. Dieser Transform-Knoten ist der im VRML-Szenegraphen enthaltenen Geometrie übergeordnet. Die Positionswerte des eventOut Feldes werden dabei an das Feld translation übertragen. Die Position des Objektes ist dadurch solange an den sich im 3D-Fenster bewegenden Mauszeiger gebunden, wie der Event-Handler onMousedown aktiv ist bzw. Werte vom PlaneSensor an den dazugehörigen Transform-Knoten geliefert werden.

6.5.7 Abfrage von Objektinformationen

Dieses Modul stellt die in Kapitel 6.3.1 beschriebene Funktionalität zur Abfrage von Objektinformationen durch Klicken auf das Objekt dar.
Das Konzept sieht für dieses Modul vor, dass die Objektinformationen in einer oder mehreren Datenbanken gespeichert sind. Um Informationen zu einem bestimmten Objekt aufzurufen, ist es deshalb notwendig, die Objekte mit einer eindeutigen ID zu versehen. Für das Projekt wurde die Möglichkeit genutzt, IDs in den Feldern der Anchor-Knoten vorzuhalten. Weiterhin sieht das Konzept vor, dem Nutzer die Möglichkeit zu geben, die gewünschten Informationskategorien für eine Datenbankabfrage selbst zu bestimmen. Dies kann über die Implementierung eines weiteren FORM-Elements in die Benutzeroberfläche realisiert werden. In diesem Formular wählt der Nutzer zunächst die gewünschten Kategorien aus, in denen er Informationen zu einem Objekt wünscht. Dann klickt er dasjenige Objekt an, zu welchem die Informationsabfrage ausgeführt werden soll. Die ausgewählten Kategorien werden dann per URL-Aufruf an ein PHP-Script übergeben, welches die entsprechenden Informationen aus einer Datenbank abfragt. Bei dieser Art von Datenbankabfrage handelt es sich um eine Standardprozedur. Der Benutzer erhält als Ergebnis ein vom Server dynamisch generiertes HTML-Dokument, welches die gewünschten Inhalte enthält.

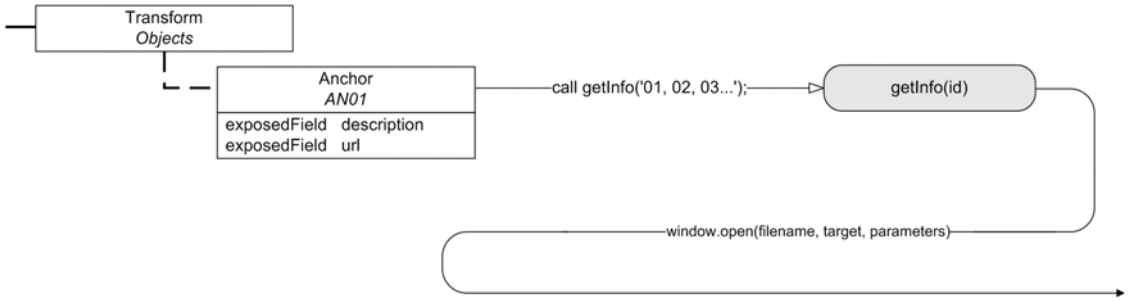

Abb. 54: Modul zur Abfrage von Objektinformationen (eigener Entwurf und Darstellung)

Da es sich bei der beschriebenen Abfrage von Informationen aus Datenbanken um eine häufig angewandte Standardmethode handelt (THEIS 2006, 184 ff.), wurde auf deren vollständige Implementierung in der jetzigen Version des Systems verzichtet. Um für Demonstrationszwecke den Vorgang simulieren zu können, werden im Demonstrator entsprechende Anfragen zu einem Objekt auf ein statisches, vorgefertigtes HTML-Dokument gelenkt, welches Informationen zu vorausgewählten Kategorien enthält (Abb. 30). Die Funktionsweise dieses Moduls ist in Abbildung 54 dargestellt. Bei Ak-

tivieren des Anchor-Knotens wird die im HTML-Dokument referenzierte JavaScript-Funktion getInfo(id) aufgerufen. Deren Parameter id wertet die beim Aufruf der Funktion übergebenen Werte aus. Dann greift das JavaScript auf das Window-Objekt des Browsers zu und öffnet über dessen Methode open ein neues Browserfenster mit den gewünschten Informationen über das vom Benutzer angeklickte Objekt.

6.5.8 Positions- und Koordinatenanzeige

Dieses Modul liefert die in Kapitel 6.3.5 beschriebene Funktionalität, welche dem Benutzer die aktuelle Position des Avatars auf einer Karte sowie in Gauß-Krüger-Koordinaten anzeigt. Ferner werden dem Nutzer durch dieses Modul Informationen über die aktuelle Höhe des Avatars sowie den gültigen Nordwinkel zur Verfügung gestellt. Die Anzeige des eingestellten Sichtfelds wird zumindest teilweise durch dieses Modul gesteuert.
Abbildung 55 stellt den Funktionsablauf dieses Moduls dar. Die Abbildung stellt in der linken oberen Ecke diejenigen Knoten dar, die innerhalb des VRML-Dokuments root.wrz für die Modulfunktion verantwortlich sind. In der rechten unteren Ecke sind diejenigen Elemente abgebildet, die im HTML-Dokument index.html für die Funktion relevant sind. Die relative Anordnung der Knoten im Ausschnitt des VRML-Szenegraphen bzw. der Elemente im Ausschnitt des DOM-Baum entspricht denen in Abbildung 44 bzw. 46. Im Übrigen gelten für die Abbildung und die nachfolgende Beschreibung dieselben Vorbemerkungen wie in den vorangegangenen Kapiteln.
Das Modul wird nicht durch den Benutzer, sondern automatisiert durch den Script-Knoten *SCStarter* in Betrieb gesetzt. Dieser Script-Knoten enthält die spezielle VRMLScript-Funktion initialize, die aufgerufen wird, sobald alle Knoten der VRML-Datei im ActiveX Control auf Clientseite geladen sind. Ist dies der Fall, dann ruft die Funktion initialize das im HTML-Dokument index.html referenzierte JavaScript-Programm starter() auf. Diese Funktion blendet daraufhin in einem ersten Schritt die für die Interaktion mit der Szene notwendigen Elemente der graphischen Benutzeroberfläche ein. Diese auf den ersten Blick kompliziert erscheinende Methode des Aufrufs ist notwendig, um sicherzustellen, dass alle Knoten des VRML-Szenegraphen, auf die das HTML-Dokument zugreift, auch tatsächlich bereits zur Verfügung stehen. In einem zweiten Schritt startet das JavaScript starter() zwei weitere Programme, die per Definition in bestimmten Zeitintervallen immer wieder aufgerufen werden. Dies sind

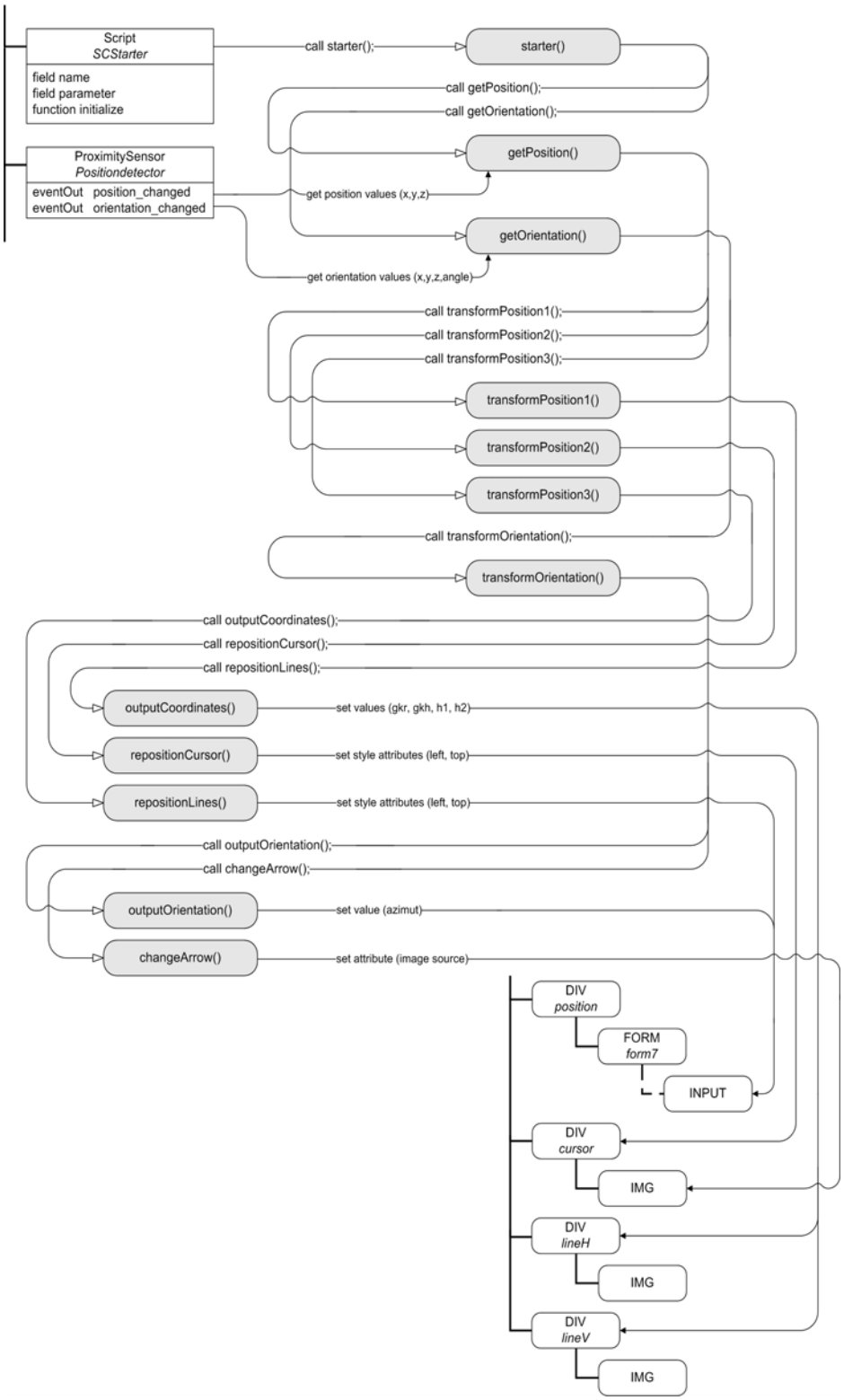

Abb. 55: Modul zur Positions- und Koordinatenanzeige (eigener Entwurf und Darstellung)

die JavaScript-Funktionen getPosition() und getOrientation(); sie werden zeitlich versetzt in Intervallen von 50 Millisekunden aktiv. Keines der JavaScript-Programme, die im Rahmen dieses Moduls aufgerufen werden, erwartet Parameter oder übergibt Werte direkt an andere Funktionen. Stattdessen werden für alle notwendigen Werte dokumentweit gültige Variablen deklariert.
Das JavaScript getPosition() führt dazu die Variablen positionX, positionY und positionZ ein. Der ProximitySensor *Positiondetector* wurde so gestaltet, dass er die gesamte 3D-Szene einschließt. Jede Bewegung des Avatars kann somit erfasst werden. Der ProximitySensor *Positiondetector* liefert über seine eventOut Felder position_changed und orientation_changed die jeweilige Position und Ausrichtung im lokalen Koordinatensystem der Szene. Die Funktion getPosition() fragt die Werte des Felds position_changed ab und legt diese in den genannten Variablen ab.
Das Programm getOrientation() führt die Variablen orientationX, orientationY, orientationZ sowie orientationAngle ein. Die Funktion greift dann auf die Werte des Felds orientation_changed des ProximitySensors *Positiondetector* zu und speichert diese in den dafür deklarierten Variablen. Die Positions- und Orientierungskoordinaten liegen wie erwähnt zu diesem Zeitpunkt als lokale Koordinaten des in der VRML-Szene gültigen Koordinatensystems vor. Um sie für die Steuerung der verschiedenen Elemente innerhalb der graphischen Benutzeroberfläche nutzen zu können, sind umfangreiche Transformationen notwendig.
Für die Transformation der Positionskoordinaten ruft das JavaScript getPosition() drei verschiedene Funktionen auf. Die Funktion transformPosition1() führt mit den über die global gültigen Variablen zur Verfügung stehenden Positionskoordinaten eine Affintransformation durch. Diese liefert als Ergebnis die Position auf Basis des Gauß-Krüger-Systems. Hierzu wird in einem ersten Schritt zu den lokalen Koordinaten das für die Daten eingeführte Offset von 347000 für den Rechtswert bzw. 547000 für den Hochwert aufgeschlagen. Dann werden beide Werte auf ganze Zahlen gerundet, was die kleinstmögliche Maßeinheit auf einen Meter beschränkt. Zwar ist es durchaus möglich, mit Nachkommastellen zu arbeiten, allerdings würden diese eine Genauigkeit implizieren, die das System aufgrund der Datengrundlage und Rundungsfehlern nicht liefern kann. Da für die Höhe kein Offset eingeführt wurde, kann dieser Wert unverändert übernommen werden. Durch eine einfache Addition wird aus der Höhe über Grund der Wert für die Höhe über Normalnull gewonnen. Die auf diese Weise transformierten Werte werden wiederum in global deklarierten Variablen abgespeichert. Die Funktion transformPosition2() führt mit den Positionskoordinaten ebenfalls eine

ebene Transformation durch. Diese liefert als Ergebnis diejenigen Koordinaten, die dem Maßstab und der Ausrichtung der in der Benutzeroberfläche integrierten Übersichtskarte zu Grunde liegen. In der Funktion wird darüber hinaus eine Reihe von bedingten Anweisungen durchgeführt. Diese fangen diejenigen Werte auf, die außerhalb der Übersichtskarte liegen. Die transformierten Werte werden ebenfalls in global gültigen Variablen abgelegt. In der Funktion transformPosition3() wird vom Prinzip her dieselbe Koordinatentransformation durchgeführt wie in der vorangegangenen. Aus gewissen technischen Gründen ist es aber für die dynamische Positionierung von Layern in einem HTML-Dokument notwendig, Offsets einzuführen. Diese sind abhängig von der Ausdehnung der anzusteuernden Layer. Da die Transformationswerte aus den beiden Funktionen zur Steuerung dreier unterschiedlicher Layer bestimmt sind, ist das jeweils notwendige Offset in der Transformation berücksichtig. Auch im JavaScript transformation3() wird durch eine Reihe bedingter Anweisungen sichergestellt, dass keine Werte generiert werden, die außerhalb der Übersichtskarte liegen. Die in dieser Funktion transformierten Werte werden wie gehabt in dokumentweit aufrufbaren Variablen abgelegt.

Die in den drei verschiedenen JavaScript-Programmen transformierten Werte werden im Anschluss an dasjenige Element der Benutzeroberfläche weitergeleitet, welches mittels der Werte gesteuert werden soll. Das JavaScript transformPosition1() ruft im Anschluss an die Koordinatentransformation die Funktion outputCoordinates() auf. Diese greift auf die verschiedenen INPUT-Elemente des Formular-Knotens *form7* innerhalb des DIV-Elementes *position* zu und belegt deren Eingabefelder mit den Werten aus der Funktion transformPosition1(). Dies sind Rechtswert, Hochwert, Höhe über Normalnull und Höhe über Grund. Das JavaScript transformPosition2() startet nach der Transformation der Koordinaten die Funktion repositionCursor(). Diese greift auf das DIV-Element *cursor* zu und belegt dessen Style-Attribute left und top mit den transformierten Werten. Diese Style-Attribute sind für die Positionierung des Cursors auf der Übersichtskarte zuständig. Die JavaScript-Funktion transformPosition3() schließlich aktiviert nach erfolgter Neuberechnung der Koordinaten das JavaSript-Programm repositionLines(). Dieses belegt analog die gleichen Style-Attribute bei den beiden DIV-Elementen *lineH* und *lineV*. Dadurch wird die horizontale bzw. vertikale gelbe Linie auf der Übersichtskarte verschoben.

Für die Transformation der Orientierungskoordinaten ruft das JavaScript-Programm getOrientation() die Funktion transformOrientation() auf. Wie in Kapitel 6.1.6 angesprochen, werden alle Drehungen in VRML als Achs-Winkel Drehungen durchgeführt. Auch die Ausrichtung des aktuellen Viewpoints wird so definiert. Die Parameter, die bei der Achs-Winkel Drehung zur Lagedefinition eines Objektes im dreidimensionalen Raum genutzt werden, lassen keinen direkten Zugriff auf das Azimut zu. Das Azimut, das die auf Norden bezogene horizontale Richtung zu einem Punkt in einem Winkel definiert, muss durch eine Umformung der Parameter der Achs-Winkel Drehung berechnet werden. Diese Umformung wird im JavaScript-Programm transformOrientation() durchgeführt. Mit Hilfe des Skalarprodukts wird dabei der Winkel zwischen zwei Vektoren im euklidischen Raum berechnet. Abbildung 56 veranschaulicht den Umformungsprozess. Der Pfeil mit der unterbrochenen Linie stellt den Vektor bzw. die Achse dar, um die ein Objekt gedreht wird. Der kreisförmige, dickere Pfeil symbolisiert den Winkel, um den ein Objekt um die genannte Achse gedreht wird. Der Vektor und der dazugehörige Winkel wurden in der JavaScript-Funktion getOrientation() abgefragt und in globalen Variablen gespeichert. In der Umformung durchlaufen nun diese Variablen eine Gleichung deren Ergebnis der Winkel a aus Abbildung 56 ist.

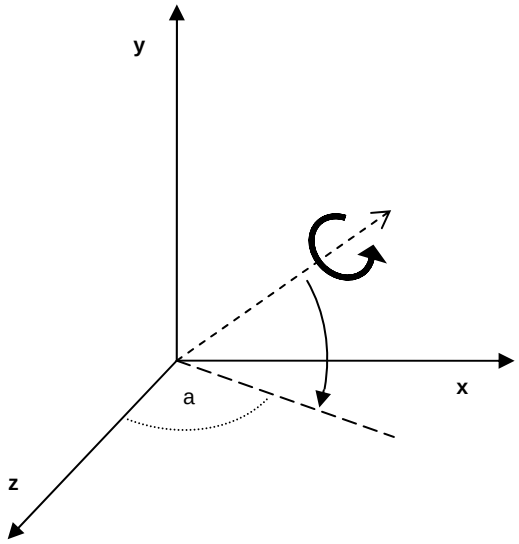

Abb. 56: Geometrische Darstellung der Umformung von Achs-Winkel Parametern (eigene Darstellung)

Geometrisch gesehen wird dabei der Vektor auf die xz-Ebene projiziert. Der Winkel a liegt zwischen dem projizierten Vektor und der z-Achse. Der Wertebereich dieses Winkels liegt zwischen -1pi und + 1pi. Über weitere Konvertierungen wird daraus schließlich der Nordwinkel (Azimut) des aktuellen Viewpoints abgeleitet. Die Grundlagen, auf denen der Umformungsprozess sowie die für dieses Modul aufgestellte Gleichung beruht, finden sich bei BRILL (2005) bzw. LEUPOLD (1990).
Die in dem JavaScript transformOrientation() transformierten Werte bzw. der daraus abgeleitete Wert für den Nordwinkel wird im Anschluss an diejenigen Elemente der Benutzeroberfläche weitergeleitet, welche mittels dieses Werts dynamisch verändert werden sollen. Dazu ruft die Transformationsfunktion nach erfolgter Umrechnung die beiden JavaScript-Programme outputOrientation() sowie changeArrow() auf.
Die Funktion outputOrientation() greift in derselben Art wie outputCoordinates() auf die verschiedenen INPUT-Elemente des Formular-Knotens *form7* innerhalb des DIV-Elementes *position* zu. Sie belegt eines deren Eingabefelder mit dem für den Nordwinkel berechneten Wert aus dem JavaScript transformOrientation(). Die Ausgabe dieses Wertes erfolgt wie in Kapitel 6.3.5 beschrieben in Winkelgrad.
In der Funktion changeArrow() durchläuft der Wert des Nordwinkels zunächst eine Reihe von bedingten Anweisungen, in denen der aktuelle Wert einer Klasse zugeordnet wird, die eine Breite von 10° hat. In Abhängigkeit der Klasse, in die der Wert fällt, wird das Attribut für die Bildquelle (image source) im IMG-Element des DIV-Knotens *cursor* verändert (Abb. 55). Dadurch zeigt der in der graphischen Benutzeroberfläche für den Cursor gewählte, rote Pfeil in diejenige Richtung, in die der Betrachter bzw. der aktive Viewpoint gerade schaut (Abb. 27).

7 Schlussbetrachtung

Die Nachfrage nach dreidimensionalen Geodaten nimmt in Industrie und Öffentlichkeit immer weiter zu (COORS & ZIPF 2005, VII ff.; GRÜNBECK 2004, 129 ff.). Der wachsende Zugriff auf kleinmaßstäbliche, raumbezogene Inhalte über seit einigen Monaten verfügbare Online-Globen wie Google Earth oder Nasa World Wind unterstreicht dies (GOO & KLEIN 2007). Die Entwicklung professioneller GIS-Lösungen für 3D-Geodaten schreitet langsam aber stetig voran. Während in vielen kommunalen Ämtern bereits am Aufbau von 3D-Stadtmodellen gearbeitet wird und einige Kommunen wie Berlin sogar schon detaillierte Stadtmodelle für Planungszwecke nutzen, bleibt der Öffentlichkeit der direkte Zugang zu dieser Art von dreidimensionalen Modellen jedoch noch weitgehend verwehrt. Dies liegt zum einen an der fehlenden Verfügbarkeit flächendeckender Modelle und zum anderen am Mangel von geeigneten Systemen, die die Daten einem größeren Publikum in zweckdienlicher Form zur Verfügung stellen. Untersuchungen belegen, dass dreidimensionale virtuelle Modelle im Vergleich zu herkömmlichen zweidimensionalen Darstellungen wesentlich besser geeignet sind, um komplexe räumliche Sachverhalte für Laien verständlich zu machen (HAMILTON ET AL. 2001, 833 ff.). Methoden zur Erhebung, Verwaltung und Präsentation dreidimensionaler Geodaten sind deshalb aktuell Gegenstand intensiver wissenschaftlicher Forschung. Was liegt also näher als ein dreidimensionales Stadtmodell eingebunden in eine virtuelle, interaktive Umgebung zu allgemeinen Informationszwecken und zur Bürgerbeteiligung in Planungsprozessen einzusetzen? Hier setzte die vorliegende Arbeit an.

Im Mittelpunkt der Arbeit standen die Entwicklung eines interaktiven, webbasierten 3D-Informationssystems für den Heidelberger Universitätscampus sowie die Generierung der dafür notwendigen dreidimensionalen Inhalte. Da die Arbeit thematisch mehrere Wissenschaftsbereiche berührt, wurde eine interdisziplinäre Betrachtungsweise gewählt. Diese vereint verschiedene Methoden und Erkenntnisse aus Geographie, Planungswissenschaft, Architektur, historischer Stadtentwicklung, Computergrafik und Informatik unter dem gemeinsamen Dach der angewandten Geographie.

Mit dem ersten Forschungsziel, der digitalen Abbildung von planerischen Prozessen und Inhalten, sollte aufgezeigt werden, welche Anknüpfungspunkte der bauliche Planungsprozess für eine Unterstützung durch dreidimensionale Visualisierung bietet. Hierfür wurden die formellen Ebenen der Bauleitplanung und Objektplanung abstrahiert und in generalisierter Form betrachtet. Es wurde ein Entwurf geleistet, der Planung als iterativen Kommunikationsprozess verschiedener Akteure versteht. Aus dieser Konzeption wurden Anknüpfungspunkte für ein 3D-Informationssystem abgeleitet. Dies waren neben dem formalen Prozess der Öffentlichkeitsbeteiligung auch spezielle Teilprozesse bzw. Instrumente, die das klassische Konzept analoger Architekturmodelle in eine digitale Umgebung überführen. Die digitale Abbildung dieser planerischen Prozesse und Instrumente bietet vielfältige Vorteile und ist unabhängig von den konkreten Planungsinhalten vorliegender Arbeit auf andere Kommunen übertragbar.

Das zweite Forschungsziel, die Schaffung eines digitalen mehrdimensionalen Abbilds des Universitätscampus, war eine besondere Herausforderung. Durch die konsequente Berücksichtigung im Vorfeld aufgestellter spezifischer Bedingungen, konnte ein virtuelles 3D-Modell des Heidelberger Universitätscampus geschaffen werden, das unterschiedlichsten Anforderungen gerecht wird. Auf der einen Seite sind mit dem Modell anspruchsvolle, quasi fotorealistische Visualisierungen möglich, andererseits erlauben die vollautomatisch veränderbaren Parameter des Modells aber auch dessen Verwendung in einer interaktiven virtuellen Echtzeitumgebung auf Standardrechnern. Neben dem Gebäudebestand finden zukünftige und historische Planungen Eingang in das Modell. Hierdurch wird die Zeit als vierte Dimension integriert.
Es wurde eine flächendeckende geometrische Detailmodellierung des gesamten Gebäudebestandes des Universitätscampus geleistet. Die Texturierung der Geometrie basiert auf rund 1200 eigens zu diesem Zweck aufgenommenen Fassadenfotos. Um einen idealtypischen und homogenen Charakter des Modells zu erreichen, wurden die Fotos aufwendig bereinigt und perspektivisch entzerrt. Für zukünftige Vorhaben und historische Planungen wurden Pläne von Architekten bzw. historische Grund- und Aufrisse sowie klassische Architekturmodelle analysiert. Das 3D-Modell besteht insgesamt aus rund 100 Einzelgebäuden. Es enthält darüber hinaus ein Infrastrukturmodell, welches den gesamten Verkehrswege- und Grünflächenbestand abbildet. Um darüber hinaus ein Geländemodell umzusetzen, wurden markante Reliefunterschiede wie das Neckarbett aufwendig digitalisiert; für einige charakteristische Geländemerkmale wurden zudem eigene Erhebungen durchgeführt. Die an das Untersuchungsgebiet an-

grenzende Bebauung von Neuenheim und Wieblingen wurde ebenfalls digitalisiert und als generalisiertes 3D-Modell umgesetzt.
Für die dem Modell zu Grunde liegenden Daten wurde eine Prozesskette entworfen, die Erfassung und Verwaltung sämtlicher zweidimensionaler Daten in einem Geographischen Informationssystem abwickelt. Neben offiziellen Planwerken wie dem Flächennutzungs-, Bebauungs- oder Stadtplan wurden aktuelle und historische Gesamtplanungen des Universitätsbauamtes, Luftbilder des Stadtvermessungsamtes, CAD-Pläne von Architekturbüros sowie eigene Digitalisierungen in das GIS integriert. Insgesamt stehen rund 20 verschiedene Informationsebenen zur Verfügung. Die Verwaltung in einem GIS erlaubt neben entsprechenden Analysen auch die Bereitstellung der Inhalte über ein WebGIS.
Ein Film auf der beiliegenden CD zeigt das 3D-Modell im Kontext unterschiedlicher Informationsebenen.

Der konzeptionelle Entwurf des Informationssystems stellte das dritte Forschungsziel dar. Hier wurden inhaltliche, funktionale und technische Anforderungen aufgestellt, denen das Informationssystem gerecht werden sollte. Die inhaltlichen und funktionalen Ansprüche leiteten sich zu einem großen Teil aus dem ersten Forschungsziel ab. Darüber hinaus wurden ähnliche Informationssysteme auf ihren Funktionsumfang hin untersucht. Literatur, die sich mit der Gestaltung von Informationssystemen beschäftigt, wurde ausgewertet. Für die technischen Anforderungen wurden zwei Prämissen gesetzt. Zum einen sollte das System ubiquitär verfügbar sein und zum anderen sollte es sowohl in den zugrunde liegenden Technologien wie auch in den benutzten Dateiformaten auf international anerkannten Standards basieren. Mit der Wahl des Mediums Internet sowie der Auswahl von Programmiersprachen bzw. Dateiformaten, die von international anerkannten Organisationen wie der ISO oder dem W3C als Standard spezifiziert sind, wurden beide Prämissen konsequent beachtet.
Hinsichtlich des Funktionsumfangs des Informationssystems vereint das Konzept verschiedene Zielgruppen. Das eher allgemein geprägte räumliche Informationsbedürfnis von Studenten, Besuchern, Patienten der Kliniken oder generell interessierten Personen wird ebenso befriedigt wie das planungspartizipativ geprägte Informations- und Kommunikationsinteresse von Bürgern, Anwohnern und Nutzern. Als weitere Zielgruppe wurden offiziell am Planungsprozess beteiligte Akteure wie Planer und Ingenieure einbezogen. Für diese wurden spezielle Konzeptions- und Argumentationsinstrumente entworfen.

Die Implementierung des interaktiven 3D-Informationssystems, dem vierten Forschungsziel, war neben der Schaffung des multifunktionalen 3D-Modells die zweite große Herausforderung der vorliegenden Arbeit. Hier finden die Ergebnisse der übrigen Forschungsziele zusammen.
Erstmals konnte ein browserbasiertes 3D-Informationssystem realisiert werden, das konsequent auf international anerkannten Standards basiert und einen großflächigen Baubestand inklusive historischer Planungen sowie zukünftiger Vorhaben einem breiten Benutzerkreis in interaktiver Weise und in Echtzeit erschließt. Das System ist dabei vollständig in die Hyperreferenz-Struktur des Internets eingebunden. Externe Inhalte und Datenbanken können ebenso in das Informationssystem integriert werden, wie das System selbst von fremden Webseiten über Hyperlinks und unter Übergabe definierter Parameter angesprochen werden kann. Das Informationssystem basiert auf offenen Standards und bietet daher eine ganze Reihe von Vorteilen gegenüber proprietären Systemen. Die Funktionalitäten des System sind von Dritten erweiterbar. Die Weiterentwicklung der verwendeten Standards ist nicht von einzelnen kommerziellen Herstellern abhängig, deshalb werden keine Lizenzgebühren fällig. Außerdem garantiert der Aufbau auf diesen Standards die zukünftige Investitionssicherheit des Systems. Der modulare Aufbau der Systemfunktionalitäten erlaubt ein schnelles und einfaches Anpassen bzw. Erweitern einzelner Funktionsmodule ohne dabei die Gesamtarchitektur nachteilig zu beeinflussen. Für den Systemkern mit der graphischen Benutzeroberfläche und den einzelnen Funktionsmodulen wurden vom Verfasser rund 3000 Zeilen Quellcode programmiert. Zusammen mit den dreidimensionalen Inhalten beträgt das Datenvolumen des gesamten Informationssystems gerade einmal 10 Megabyte. Dies kommt einer Anwendung über das Internet ebenso entgegen wie dem Offline-Betrieb des Systems auf einem lokalen Rechner. Letzteres ist im Gegensatz zu WebGIS-Anwendungen uneingeschränkt und ohne die Notwendigkeit einer lokalen Serverinstallation möglich.
Das Informationssystem konnte bereits erfolgreich online erprobt werden. Dies wird auf der beiliegenden CD anhand eines Films demonstriert.

Es wurden vier Forschungsziele definiert, die jeweils vollständig erreicht wurden. Die Forschungsergebnisse leisten einen wichtigen Beitrag, um Forschungslücken im Bereich der interaktiven Online-Präsentation von 3D-Daten sowie der Verwendung von dreidimensionalen Stadtmodellen im Rahmen der webbasierten, planungsbezogenen Öffentlichkeitsbeteiligung zu schließen. In Bezug auf die digitale Abbildung Heidel-

bergs wird ebenfalls eine wissenschaftliche Lücke geschlossen. Mit dem 3D-Modell des Universitätscampus steht erstmals ein detailliertes, flächendeckendes digitales Abbild eines ganzen Stadtteils zur Verfügung, welches es nicht nur erlaubt, den baulichen Bestand inklusive historischer Planungen und zukünftiger Vorhaben fotorealistisch zu visualisieren, sondern auch interaktiv und online für einen breiten Nutzerkreis zur Verfügung zu stellen.

Literaturverzeichnis

ACHSTETTER, S.; FREIWALD, N.; JANY, R. (2006): Interaktives historisches Heidelberg. In: WUNDER, E.; CLEMENS, J.; GAMERITH, W.; GEBHARDT, H.; LOSSAU, J.; MARXHAUSEN, C.; SCHWAN, T. (Hrsg.): Ozeane – Äquatoriales Afrika – Rund um das Dach der Welt – Besonderheiten der politischen Landkarte Europas. (=HGG-Journal 19+20). Heidelberg. 235-237.

ALEXA, M. (2006): Vereinfachung polygonaler Netze. http://cg.cs.tu-berlin.de/lehre06ss/cg2/slides/cg2-mesh2.pdf. Abruf am 02.12.06.

ALTENSCHMIDT, U. (1999): COM – Component Object Model. http://www.littlesoft.com/pdf/com.pdf. Abruf am 02.03.07.

AMES, A. L.; NADEAU, D. R.; MORELAND, J. L. (1997): VRML 2.0 Sourcebook. Wiley. New York.

ANDREW, R.; SHAFER, D. (2006): CSS – Anspruchsvolle Websites mit Cascading Stylesheets. dpunkt-Verlag. Heidelberg.

ASCHE, H.; HERRMANN, C. (2003): Web.Mapping 2 – Telekartographie, Geovisualisierung und mobile Geodienste. Wichmann. Heidelberg.

BACHMANN, M.; HELLMEIER, J.; ALBERT. J. (2003): Zielgruppen und Anwendungen für Digitale Stadtmodelle und Digitale Geländemodelle Erhebungen im Rahmen der Arbeitsgruppe „Anwendungen und Zielgruppen“ der SIG3D im Rahmen der Initiative GDI-NRW. http://www.ikg.uni-bonn.de/fileadmin/sig3d/pdf/ Tabelle_Anwendungen_Zielgruppen.pdf. Abruf am 05.06.06.

BAKER, M. J. (2007): Transformation Theory. http://euclideanspace.com/maths/geometry/transform/index.htm. Abruf am 02.03.07.

BALZERT, H. (2003): HTML, XHTML & CSS für Einsteiger – Statische Websites dynamisch erstellen. W3L-Verlag. Herdecke.

BARTELME, N. (2005): Geoinformatik – Modelle, Strukturen, Funktionen. Springer. Heidelberg.

BAUER, P.; TEUFEL, D. (2004): Möglichkeiten zur besseren Anbindung des Universitätscampus im Neuenheimer Feld und zur Entlastung der Hauptstraßen in Handschuhsheim. UPI-Institut. Heidelberg.

BAUER, W.; MOHL, H.-U. (2005): Das 3D-Modell der Landeshauptstadt Stuttgart. In: COORS, V.; ZIPF, A. (Hrsg.): 3D-Geoinformationssysteme – Grundlagen und Anwendungen. Wichmann. Heidelberg. 265-278.

BECKER, D.; SATISH, J. (2003): Für Planung und Marketing – Erstellung eines 3-D-Stadtmodells unter Nutzung vorhandener Daten. In: Geo-Informations-Systeme/ GeoBIT 8 (2003). 19-21.

BECKER, H. (2002): Stadtbaukultur: Modelle, Workshops, Wettbewerbe – Verfahren der Verständigung über die Gestaltung der Stadt. Schriften des Deutschen Instituts für Urbanistik. Bd.88. Deutsches Institut für Urbanistik. Berlin.

BERLINER SENATSVERWALTUNG FÜR WIRTSCHAFT, ARBEIT UND FRAUEN (2006): Pressemeldung vom 18.05.06. http://www.berlin.de/landespressestelle/archiv/2006/05/18/ 40301/index.html. Abruf am 18.05.06

BERLIN PARTNER GMBH (2007): Berlin in der 3. Dimension erleben. http://www.3d-stadtmodell-berlin.de/3d/C/seite1.jsp?nav1=open. Abruf am 17.03.07.

BERNERS-LEE, T.; CONNOLY, D. (1995): HyperText Markup Language Specification 2.0. http://www.w3.org/MarkUp/html-spec/. Abruf am 02.03.07

BILL, R. (1999a): Grundlagen der Geo-Informationssysteme – Hardware, Software und Daten. Bd.1. Wichmann. Heidelberg.

BILL, R. (1999b): Grundlagen der Geo-Informationssysteme – Analysen, Anwendungen und neue Entwicklungen. Bd.2. Wichmann. Heidelberg.

BISCHOF, A.; SELLE, K.; SINNING, H. (2005): Informieren, Beteiligen, Kooperieren – Kommunikation in Planungsprozessen. Dortmunder Vertrieb für Bau- und Planungsliteratur. Dortmund.

BORN, G. (2000): HTML – Kompendium. Markt & Technik. München.

BRAIL, R. K.; KLOSTERMANN, R. E. (Eds.) (2001): Planning Support Systems – Integrating Geographic Information Systems, Models and Visualization Tools. ESRI Press. Redlands.

BRAUN, H. (2007): Webstandards im Wandel. In: C'T – Magazin für Computertechnik. 1/07. Heise Verlag. Hannover. 162-169.

BRILL, M. (2005): Mathematik für Informatiker. Hanser. München.

BRENNER C.; KOLBE, T. (2005): Neue Perspektiven – Wie 3D-Stadtmodelle erstellt und verwendet werden. In: C'T – Magazin für Computertechnik. 15/2005. 106-111.

BRODLIE, K.; EL-KHALILI, N. (2002): Web-based virtual environments. In: FISHER, P.; UNWIN, D. (Eds.) (2002): Virtual reality in geography. Taylor & Francis. London. 35-46.

BRODLIE, K.; DYKES, J.; GILLINGS, M.; HAKLAY, M. E.; KITCHIN, R.; KRAAK, M.-J. (2002): Geography in VR. In: FISHER, P.; UNWIN, D. (Eds.) (2002): Virtual reality in geography. Taylor & Francis. London. 9-16.

BRÜGGE, B., DUTOIT, A. H. (2004): Objektorientierte Softwaretechnik. Pearson. München.

BUCHHOLZ, H.; BOHNET, J.; DÖLLNER, J. (2005): Smart Navigation Strategies for Virtual Landscapes. In: BUHMANN, E.; PAAR, P.; BISHOP, I.; LANGE, E. (Eds.): Trends in Real Time Landscape Visualization and Participation – Proceedings at Anhalt University of Applied Science 2005. Wichmann. Heidelberg. 115-123.

BUNDESANZEIGER (Hrsg.) (2004): Bundesgesetzblatt. Jg. 2004. Teil I. Nr. 31. Bundesanzeiger Verlag. Köln.

BUNGARTZ, H.-J.; GRIEBEL, M.; ZENGER, C. (2002): Einführung in die Computergrafik – Grundlagen, geometrische Modellierung, Algorithmen. Vieweg. Braunschweig.

CARTWRIGHT, W. ; PETERSON, M.P.; GARTNER, G (1999): Multimedia Cartography. Springer. Berlin.

CASTRO, E. (2000): HTML 4. Markt&Technik. München.

CHEN, Y.; KNAPP, S. (2006): VEPS – Virtual Environmental Planning System – First Steps towards a web-based 3D-planning and participation tool. In: SCHRENK, M. (Hrsg.): CORP 2006 – Beiträge zum 11. Symposion zur Stadtplanung und Regionalentwicklung in der Informationsgesellschaft. Selbstverlag des Vereines CORP – Competence Center for Urban and Regional Development. Wien 275-285.

CHRISTIANSEN, T., ERB, W.-D. (2002): GIS. In: BRUNOTTE, E.; GEBHARDT, H.; MEURER, M.; MEUSBURGER, P.; NIPPER, J. (Hrsg.) (2002): Lexikon der Geographie. – CD-Rom. Spektrum Akademischer Verlag. Heidelberg.

CIESLIK, B. (2003): Hamburg in der dritten Dimension. In: Zeitschrift für Geodäsie, Geoinformation und Landmanagement. 4/2003. 128.Jg. 254-260.

COORS, V.; ZIPF, A. (2005): Einführung 3D-Geoinformationssysteme. In: COORS, V.; ZIPF, A. (Hrsg.): 3D-Geoinformationssysteme – Grundlagen und Anwendungen. Wichmann. Heidelberg. VII-XII.

DÄSSLER, R. (1999): VRML. BHV. Kaarst.

DECKER, F.; RUNDE, C. (2003): Fluchtpunkt – VRML-Plug-ins auf dem Prüfstand. In: iX- Magazin für professionelle Informationstechnik. 10/2003. 68-71.

DEUTSCHER STÄDTETAG (2005): 3D-Stadtmodelle – Orientierungshilfe des Städtetages NRW. Mitteilungen des Deutschen Städtetages. Folge 4. Nr.159. Jg.60.

DICKMANN, F. (2004): Einsatzmöglichkeiten neuer Informationstechnologien für die Aufbereitung und Vermittlung geographischer Informationen – Das Beispiel kartengestützte Online-Systeme. Göttinger Geographische Abhandlungen. Bd.112

DIEHL, S. (2001): Distributed Virtual Worlds – Foundations and Implementation Techniques Using VRML, Java and CORBA. Springer. Berlin.

DORFFNER, L.; ZÖCHLING, A. (2004): Das 3D-Stadtmodell von Wien – Grundlage für Planungsaufgaben und Visualisierungen. In: STROBL, J.; BLASCHKE, T.; GRIESEBNER, G. (Hrsg.): Angewandte Geoinformatik 2004 – Beiträge zum 16. AGIT-Symposium Salzburg. Wichmann. Heidelberg. 90-99.

DÜLFFER, J.; THIES, J.; HENKE, J. (Hrsg.) (1978): Hitlers Städte – Baupolitik im Dritten Reich – Eine Dokumentation. Boehlau. Köln.

EBERT, D. S.; MUSGRAVE, F. K.; PEACHEY, D.; PERLIN, K. WORLEY, S. (2003): Texturing & Modeling. Morgan Kaufmann Publishers. London.

ECMA (1997): ECMAScript Language Specification. ECMA-262. http://www.ecma-international.org/publications/standards/Ecma-262.htm. Abruf am 02.03.07.

FERG, S. (2006): Event-Driven Programming: Introduction, Tutorial, History. http://eventdrivenpgm.sourceforge.net/event_driven_programming.pdf. Abruf am 02.03.07.

FISCHER, J. (2004): Verkehrstelematik in Großstädten als Alternative zum Neubau von Verkehrswegen am Beispiel des Neuenheimer Feldes in Heidelberg. Diplomarbeit. Institut für Verkehrswesen. Universität Kaiserslautern.

FISCHER, M.; ZIPF, A. (2006): Mainz Mobil 3D – ein Navigationssystem für PDAs unter Nutzung des OGC Web3D Service. http://webmap.geoinform.fh-mainz.de/gdi3d/paper/AGIT.mainzMobile3D.pdf. Abruf am 02.03.07

FLANAGAN, D. (2002): JavaScript – Das umfassende Referenzwerk. O'Reilly. Köln.

FLANAGAN, D. (2006): JavaScript – The Definitive Guide. O'Reilly. Sebastopol.

FLECHTNER, I. (2000): Architektur im Nationalsozialismus. Diplomarbeit. Geographisches Institut der Universität Heidelberg.

FRAHM, T.; GNEST, H. (2004): Regionalplanung im Internet – Bundesweite Status-quo-Analyse und Empfehlungen für die Öffentlichkeitsarbeit und Beteiligung. In: SCHRENK, M. (Hrsg.): CORP 2004 – Beiträge zum 9. Symposion zur Rolle der Informationstechnologie in der Stadt- und Raumplanung sowie zu den Wechselwirkungen zwischen realem und virtuellem Raum. Selbstverlag des Vereines CORP – Competence Center for Urban and Regional Development. Wien. 421-428.

FREIWALD, N. (2003): Heidelberger Universitätsgebiet – Im Neuenheimer Feld. Diplomarbeit. Geographisches Institut der Universität Heidelberg.

FREIWALD, N.; JANY, R. (2005): Dateiformate für vektorbasierte 3D-Geodaten. In: COORS, V. ; ZIPF, A. (Hrsg.): 3D-Geoinformationssysteme – Grundlagen und Anwendungen. Wichmann. Heidelberg. 142-158.

FREIWALD, N.; GÖBEL, R.; JANY, R. (2006): Dreidimensionale Modellierung von Geoobjekten mit GIS und CAD. Heidelberger Geographische Bausteine. H. 17. Selbstverlag des Geographischen Instituts der Universität Heidelberg. Heidelberg.

FRITSCH, D. (2003): 3D Building Visualization – Outdoors and Indoors Applications. In: Geo-Informations-Systeme/ GeoBit 16 (2003). 26-32.

FRÜH, C.; ZAKHOR, A. (2003): Constructing 3D City Models by Merging Ground-Based and Airborne Views. Proceedings. In: Computer Graphics and Applications. Vol. 23. Issue 6. 52-61.

FUH, J. Y. H.; LI, W. D. (2005): Advances in collaborative CAD – the state of the art. In: Computer-Aided Design. Vol. 37. Issue 5. 571-581.

FUHRMANN, S.; MACEACHREN, A. M. (2001): Navigation in desktop-basierten geovirtuellen Welten. In: Kartographische Nachrichten. Vol.51. Nr.3. 132-142.

GALITZ, W. O. (2002): The Essential Guide to User Interface Design: An Introduction to GUI Design Principles and Techniques. Wiley. New York

GAMPERL, J. (2007): AJAX – Grundlagen, Frameworks, APIs. Galileo Press. Bonn.

GENSTORFER, H. (2005): 3D für Fahrzeugnavigationssysteme. Diplomarbeit. Institut für Telekommunikation und Medien. Fachhochschule St. Pölten.

GÖBEL, R. (2006): Analyse und Visualisierung dreidimensionaler Geodaten – Ein Methodenvergleich. Diplomarbeit. Geographisches Institut der Universität Heidelberg.

GÖBEL, R. (2007): Dreidimensionale Stadtmodelle - Offene Fragen und mögliche Lösungsansätze. Laufendes Dissertationsprojekt. Geographisches Institut der Universität Heidelberg.

GOLDFARB, C. F. (1993): The SGML Handbook. Clarendon. Oxford.

GOO, S. K.; KLEIN, A. (2007): Google searches for Government Work. http://www.washingtonpost.com/wp-dyn/content/article/2007/02/27/AR2007022701541.html. Abruf am 28.02.07.

GOODMAN, D. (2006): Dynamic HTML - The Definitive Reference. O'Reilly. Sebastopol.

GOOGLE (2006): Pressemeldung vom 12.06.06. http://www.google.com/press/pressrel/geoday.html. Abruf am 12.06.06.

GORMSEN, E. (1981): Stadt und Universität in Heidelberg. In: FRICKE, W.; GORMSEN, E. (Hrsg.): Heidelberg und der Rhein-Neckar-Raum. Heidelberger Geographische Arbeiten. H.46. Heidelberg. 111-154.

GRABMÜLLER, M. (2003): Multiparadigmen-Programmierpsprachen. http://uebb.cs.tu-berlin.de/~magr/pub/Multiparadigm-TR-2003-15.pdf. Abruf am 02.03.07.

GRAU, O. (2001): 3D-Entwicklung mit VRML und Alternativen. In iX – Magazin für professionelle Informationstechnik. 5/2000. 110.

GRIESBACH, D.; MAISANT, M. (1986): Die Chirurgische Klinik. In: RIEDL, P. A. (Hrsg.): SEMPER APERTUS – Sechshundert Jahre Ruprecht-Karls-Universität Heidelberg 1386-1986. Bd.5. Heidelberg. 498-513.

GRIGORIEVA, N. (2005): 3D-Visualisierung von VRML-Daten. Masterarbeit im Fachbereich Geoinformatik der Fachhochschule Mainz. http://www.geoinform.fh-mainz.de/diplomarbeiten/M019/. Abruf am 02.03.07.

GRÖGER, G.; KOLBE, T. H. (2005): 3D-Stadtmodellierung auf Basis internationaler GI-Standards. . In: COORS, V.; ZIPF, A. (Hrsg.): 3D-Geoinformationssysteme – Grundlagen und Anwendungen. Wichmann. Heidelberg. 234-249.

GRÜNBECK, E. (2004): Die dritte Dimension im Kataster – Eine Herausforderung für das amtliche Vermessungswesen. In: Mitteilungen des DVW-Bayern e.V. 2/2004. 129-140.

GÜLCH, E. (2005): Erfassung von 3D-Daten mit Digitaler Photogrammetrie. In: COORS, V.; ZIPF, A. (Hrsg.): 3D-Geoinformationssysteme – Grundlagen und Anwendungen. Wichmann. Heidelberg. 4-25.

HAALA, N. (2005): Laserscanning zur dreidimensionalen Erfassung von Stadtgebieten. In: COORS, V.; ZIPF, A. (Hrsg.): 3D-Geoinformationssysteme – Grundlagen und Anwendungen. Wichmann. Heidelberg. 26-38.

HAIST, J.; ETZ, M. (2005): CityServer3D. In: Computer Graphik Topics – Reports on Computer Graphics. 1/2005. Vol. 17. 16-18.

HAMILTON, A.; TRODD, N.; ZHANG, X.; FERNANDO, T.; WATSON, K. (2001): Learning trough Visual Systems to enhance the Urban Planning Process. In: Environment and Planning B: Planning and Design. Vol.28(6). 833-845.

HANGARTER, E. (1999): Bauleitplanung. Werner. Düsseldorf.

HASE, H. L. (1997): Dynamische virtuelle Welten mit VRML 2.0 – Einführung Programme und Referenz. dpunkt Verlag. Heidelberg.

HIRSEMANN, T. (2003): Anwendungsgrundlagen Internet & HTML. SPC TEIA Lehrbuch Verlag. Berlin.

ISO (1986): Information Processing – Text And Office Systems – Standard Gerneralized Markup Language (SGML). ISO 8879:1986. http://www.iso.ch/iso/en/CatalogueDetailPage.CatalogueDetail?CSNUMBER=16387&ICS1=35&ICS2=240&ICS3=30. Abruf am 02.03.07.

ISO (1997a): Space Data And Information Transfer Systems – ASCII. ISO 14962:1997. http://www.iso.ch/iso/en/CatalogueDetailPage.CatalogueDetail?CSNUMBER=26096&ICS1=49&ICS2=140&ICS3=. Abruf am 02.03.07.

ISO (1997b): Information Technology − Computer Graphics and Image Processing − The Virtual Reality Modeling Language (VRML) − Part 1: VRML 97 Functional specification and UTF-8 encoding. ISO/IEC 14772-1:1997. http://www.iso.ch/iso/en/CatalogueDetailPage.CatalogueDetail?CSNUMBER=25508. Abruf am 02.03.07.

ISO (1998): Information Technology – ECMAScript Language Specification. ISO/IEC 16262:1998. http://www.iso.org/iso/en/CatalogueDetailPage.CatalogueDetail?CSNUMBER=33835&ICS1=35&ICS2=60&ICS3=. Abruf am 02.03.07.

ISO (2000): Information Technology – Document Description and Processing Languages – HyperText Markup Language (HTML). ISO/IEC 15445:2000. http://www.iso.ch/iso/en/CatalogueDetailPage.CatalogueDetail?CSNUMBER=27688&ICS1=35&ICS2=240&ICS3=30. Abruf am 02.03.07

ISO (2002): Information Technology − Computer Graphics and Image Processing − The Virtual Reality Modeling Language (VRML) − Part 2: External authoring interface (EAI), ISO/IEC 14772-2:2002. http://www.iso.ch/iso/en/CatalogueDetailPage.CatalogueDetail?CSNUMBER=30893. Abruf am 02.03.07.

ISO (2002b): Information Technology − Computer Graphics and Image Processing − The Virtual Reality Modeling Language (VRML) − Part 1: VRML 97 Functional Specification and UTF-8 encoding − Amendment 1, ISO/IEC 14772-1/Amend.1. http://www.iso.org/iso/en/CatalogueDetailPage.CatalogueDetail?CSNUMBER=25508. Abruf am 02.03.07.

ISO (2005a): Information Technology − Computer Graphics and Image Processing − Extensible 3D (X3D) Encodings − Part 1: Extensible markup language (XML) encoding, ISO/IEC 19776-1:2005. http://www.iso.org/iso/en/CatalogueDetailPage.CatalogueDetail?CSNUMBER=33914&ICS1=35&ICS2=140&ICS3=. Abruf am 02.03.07.

ISO (2005b): Information Technology − Computer Graphics and Image Processing − Extensible 3D (X3D) Encodings − Part 2: Classic VRML encoding, ISO/IEC 19776-2:2005. http://www.iso.org/iso/en/CatalogueDetailPage.CatalogueDetail?CSNUMBER=38018&ICS1=35&ICS2=140&ICS3=. Abruf am 02.03.07.

JACKSON, M. A. (1979): Grundsätze des Programmentwurfs. Toeche-Mittler. Darmstadt.

JANY, R. (2005): Heidelberg im 17. Jahrhundert. Dissertation. Geographisches Institut der Universität Heidelberg.

JÖST, M. (2000): WebGIS – Touristeninformationssystem für die Stadt Heidelberg. Systemarchitektur und -kommunikation am Beispiel eines Tourenplanungsmoduls. Staatsexamensarbeit. Geographisches Institut. Universität Heidelberg.

KERSCHNER, M. (2005): Die Nutzung von 3D-Stadtmodellen auf GIS-Arbeitsplätzen. In: STROBL, J.; BLASCHKE, T.; GRIESEBNER, G. (Hrsg.): Angewandte Geoinformatik 2005 – Beiträge zum 17. AGIT-Symposium Salzburg. Wichmann. Heidelberg. 336-341.

KERSKEN, S. (2003): Kompendium der Informationstechnik. Galileo Press. Bonn.

KIRCHHOF, P.; SCHMIDT-AßMANN, E. (1990): Staats- und Verwaltungsrecht – Bundesrepublik Deutschland. Müller. Heidelberg.

KLOSS, J.; ROCKWELL, R.; KORNEL, S. (Hrsg.) (1998): VRML 97 – Der neue Standard für interaktive 3D-Welten im World Wide Web. Addison-Wesley. Bonn.

KOCH, S. (2006): JavaScript - Einführung, Programmierung und Referenz. dpunkt Verlag. Heidelberg.

KOCH, W. (1991): Kleine Stilkunde der Baukunst. Mosaik-Verlag. München.

KÖLMEL, K. (1952): Die Raumnot der Universität Heidelberg und der Generalbebauungsplan 1950. In: HINZ, G. (Hrsg.): Ruperto-Carola – Mitteilungen der Vereinigung der Freunde der Studentenschaft der Universität Heidelberg e.V. Jg.4. Bd.6. Heidelberg. 142-149.

KOHLHAAS, A.; BERTRAM, O. (2003): Nah an der Realität – 3D-Stadtmodelle in Architektur-CAAD-Software. In: Geo-Informations-Systeme/ Geobit 11 (2003). 16-18.

KOLKMANN, O. (2005): An Introduction to the Domain Name System. http://www.nlnetlabs.nl/downloads/DNSforPolicyMakers.pdf. Abruf am 17.11.06

KRÄMER, K. H. (1986): Die bauliche Entwicklung der Universität Heidelberg seit 1803. In: RIEDL, P. A. (Hrsg.): SEMPER APERTUS – Sechshundert Jahre Ruprecht-Karls-Universität Heidelberg 1386-1986. Bd.5. Heidelberg. 5-48.

LAMPRECHT, F. (2002): 3D für das Web – Grundlagen und Beispiele für Flash, Director, DHTML und Java. Addison-Wesley. München.

LANDESARCHIV BADEN WÜRTTEMBERG (2007): Zur Person Otto Wacker. http://www.landesarchiv-bw.de/praesmodelle/fricke3/p-wacker.htm. Abruf am 02.03.07.

LANDESVERMESSUNGSAMT BADEN-WÜRTTEMBERG (2002): TOP 25 Baden-Württemberg Version 2.0. Landesvermessungsamt Baden-Württemberg. Stuttgart.

LANGE, E. (1999): „Von der analogen zur GIS-gestützten 3D-Visualisierung bei der Planung von Landschaften. In: Geo-Informations-Systeme 2. 29-37.

LEA, R.; MATSUDA, K.; MIAYASHITA, K. (1996): Java for 3D and VRML Worlds. New Riders Publishing. Indianapolis.

LE HORS, A.; LE HEGARET, P.; WOOD, L; NICOL, G.; ROBIE, J.; CHAMPION, M.; BYRNE, S. (Eds.) (2007): Document Object Model (DOM) Level 3 Core Specification. http://www.w3.org/TR/2004/REC-DOM-Level-3-Core-20040407. Abruf am 02.03.07.

LEUPOLD, W. (1990): Analytische Geometrie, Vektorrechnung und Infinitesimalrechnung. Fachbuch-Verlag. Leipzig.

LIEBIG, W. (1999): Desktop-GIS mit ArcView GIS. Wichmann. Heidelberg

LIEBSCHER, J. (2002): Vom 2D-GIS zum 3D-GIS. In: BILL, E.; SEUß, R.; SCHILCHER, M. (Hrsg.): Kommunale Geo-Informationssysteme. Wichmann. Heidelberg. 344-358.

LINKE, M.; WINKLER, P. (1999): Computerlexikon. Markt&Technik. München.

LUBKOWITZ, M. (2005): Webseiten programmieren und gestalten. Galileo Press. Bonn.

LÜHRS, R. (2006): E-Partizipation – Ein unverzichtbarer Bestandteil ausgewogener E-Governmentstrategien. In: FRAUNHOFER E-GOVERNMENT ZENTRUM (Hrsg.): Monitoring eGovernment & Verwaltungsmodernisierung 2006/2007. Wegweiser-Verlag. Berlin. 130-131.

LURZ, M.; VOGT, D. (1990): Neuenheim im Wandel – eine Sozialgeschichte in Bildern von 1870 bis 1950. Stadtteilverein Neuenheim. Heidelberg.

MACH, R. (2000): 3D-Visualisierung. Galileo Press. Bonn.

MASLO, A.; MASLO, P.; VONHOEGEN, H. (2005): Windows XP Professional. Galileo Press. Bonn.

MATSUBA, S. N., ROEHL, B. (1996): VRML – Das Kompendium. Markt & Technik. Haar.

MEINEL, C.; SACK, H. (2004): WWW – Kommunikation, Internetworking, Webtechnologien. Springer. Berlin.

MEYER, B. (1992): Object Oriented Software Construction. Prentice-Hall. New York.

MEYER, E. (2006): CSS – The Definitive Guide. O' Reilly. Sebastopol.

MICROSOFT (2007a): Microsoft Java Virtual Machine Support. http://www.microsoft.com/mscorp/java/. Abruf am 02.03.07.

MICROSOFT (2007b): Component Object Model – COM. http://www.microsoft.com/com/default.mspx. Abruf am 02.03.07.

MICROSOFT (2007c): ActiveX Controls. http://activex.microsoft.com/activex/activex/. Abruf am 02.03.07.

MICROSOFT CORPORATION (Hrsg.) (1995): Die Windows-Oberfläche – Leitfaden zur Softwaregestaltung. Microsoft Press Deutschland. Unterschleißheim.

MIEDER, W. (1989): Ein Bild sagt mehr als tausend Worte - Ursprung und Überlieferung eines amerikanischen Lehnsprichworts. In: MIEDER, W. (Hrsg.): Proverbium: Yearbook of International Proverb Scholarship. Bd.6. DeProverbio. Vermont. S. 25-37.

MUCHMORE, M. W. (2006): 3D-Online – Browser Plugins and More. http://www.extremetech.com/article2/0,1697,2041558,00.asp. Abruf am 02.03.07.

MÜLLER, B. (1998): Architekturführer Heidelberg: Bauten um 1000-2000. Braus. Heidelberg.

MÜLLER, H. (2005): Funknetzplanung als Anwendungsgebiet für 3D-GIS. In: COORS, V.; ZIPF, A. (Hrsg.): 3D-Geoinformationssysteme – Grundlagen und Anwendungen. Wichmann. Heidelberg. 392-404.

MÜLLER, W. (2005): Grundlagen der 3D-Computergrafik. In: Coors, V.; Zipf, A. (Hrsg.): 3D-Geoinformationssysteme – Grundlagen und Anwendungen. Wichmann. Heidelberg. 184-201.

MÜNZ, S. (2007): SELFHTML Version 8.1.2. http://de.selfhtml.org/index.htm. Abruf am 02.03.07.

MÜNZ, S.; NEFZGER, W. (2002): HTML – Die Profireferenz. München.

NACHBARSCHAFTSVERBAND HEIDELBERG-MANNHEIM (2006): Flächennutzungsplan 2015-2020. http://www6.mannheim.de/nvhdma/flaechennutzungsplan/flaechennutzungsplan.htm. Abruf am 09.10.06.

NÄGELKE, H.-D. (2000): Hochschulbau im Kaiserreich – Historische Architektur im Prozess bürgerlicher Konsensbildung. Ludwig. Kiel.

NIELSEN, J., LORANGER H. (2006): Web Usability. Addison-Wesley. München.

OGC (2005): Open Geospatial Consortium – Web 3D Service. http://portal.opengeospatial.org/files/index.php?artifact_id=8869. Abruf am 02.03.07.

OGC (2006): Open Geospatial Consortium – Candidate OpenGIS CityGML Implementation Specification (City Geography Markup Language). http://portal.opengeospatial.org/files/index.php?artifact_id=16675. Abruf am 05.01.07.

OVERBECK, H. (1963): Die Stadt Heidelberg und ihre Gemarkung im Spiegel der Wandlungen ihrer Funktionen. In: PFEIFFER, G.; GRAUL, H.; OVERBECK, H. (Hrsg.): Heidelberg und die Rhein-Neckar-Lande. Keyser. Heidelberg. 80-109.

PARALLELGRAPHICS (2007a): Cortona VRML Client. http://www.parallelgraphics.com/products/cortona/. Abruf am 02.03.07

PARALLELGRAPHICS (2007b): Cortona SDK Contents – Programmer´s Reference. http://www.parallelgraphics.com/products/sdk/structure/. Abruf am 02.03.07

PARALLELGRAPHICS (2007c): Documentation for VRML Automation. http://www.parallelgraphics.com/products/sdk/structure/documentation/. Abruf am 02.03.07

PETERSON, M. P. (2003): Webmapping at the Start of the new Millenium – state of the art. In: ASCHE, H.; HERRMANN, C. (2003): Web.Mapping 2 – Telekartographie, Geovisualisierung und mobile Geodienste. Wichmann. Heidelberg. 3-18.

PFÄFFLIN, F.; DIEGMANN, V.; STAPELFELDT, H. (2004): Großräumige Lärmmodellierung im GIS. In: STROBL, J.; BLASCHKE, T.; GRIESEBNER, G. (Hrsg.): Angewandte Geoinformatik 2004 – Beiträge zum 16. AGIT-Symposium Salzburg. Wichmann. Heidelberg. 518-523.

POMASKA, G. (2007): Web-Visualisierung mit Open Source. Wichmann. Heidelberg.

PRECHELT, L. (2000): An Empirical Comparison of, C, C++, Java, Perl, Python, Rex and Tcl For A Search/String Processing Program. http://page.mi.fu-berlin.de/ ~prechelt/ Biblio/jccpprtTR.pdf. Abruf am 02.03.07.

PREIM, B. (1999): Entwicklung interaktiver Systeme. Springer. Berlin.

PREIMESBERGER, C. (2003): Java-Trends – Scriptsprachen in vorderster Reihe. http://www.iso.org/iso/en/CatalogueDetailPage.CatalogueDetail?CSNUMBER =33835&ICS1=35&ICS2=60&ICS3=. Abruf am 02.03.07

RAGGETT, D. (2002): Adding A Touch Of Style. http://www.w3.org/MarkUp/ Guide/Style.html. Abruf am 16.03.07

RAGGETT, D.; LE HORS, A.; JACOBS, I. (Eds.) (1999): HTML 4.0 Specification. http://www.w3.org/ TR/html4/. Abruf am 02.03.07.

RIEDL, A.; KATZLBERGER, G.; TOMBERGER, H. (2002): Entwicklungs- und Modellierungsumgebungen für webbasierte Geo-Virtual Reality Applikationen – eine Gegenüberstellung. In: SCHRENK, M. (Hrsg.): CORP 2002 & GeoMultimedia02 – Tagungsband. Selbstverlag des Vereines CORP – Competence Center for Urban and Regional Development. Wien. 267-274.

ROEHL, B.; COUCH, J.; REED-BALLREICH, C.; ROHALY, T.; BROWN, G. (1997): Late Night VRML 2.0 with Java. Ziff-Davis Press. Emeryville.

RÜCKBROD, A. (1969): Das bauliche Bild der Universität im Wandel der Zeit. In: LINDE, H. (Hrsg.): Hochschulplanung – Beiträge zur Struktur- und Bauplanung. Bd.1. Werner. Düsseldorf. 24-37

RÜCKBROD, A. (1977): Universität und Kollegium – Baugeschichte und Bautyp. Wissenschaftliche Buchgesellschaft. Darmstadt.

SCHILLING, A. (2002): Integration und Visualisierung von 2D- und 3D-Geodaten in einem verteilten GIS am Beispiel virtueller Stadttouren. Diplomarbeit. Geographisches Institut der Universität Heidelberg.

SCHILLING, A.; BLECHSCHMIED, H. JASNOCH, U. (2005): Datenbank-basierte Visualisierung von Hamburg. In: COORS, V.; ZIPF, A. (Hrsg.): 3D-Geoinformationssysteme – Grundlagen und Anwendungen. Wichmann. Heidelberg. 250-264.

SCHLÜTER, O. (1998): VRML – Sprachmerkmale, Anwendungen, Perspektiven. O'Reilly. Köln.

SCHMIEDER, L. (1936): Die neuen Kliniken und Naturwissenschaftlichen Institute der Universität Heidelberg. Brausdruck. Heidelberg.

SCHMIEDER, L. (1938): Der Generalbebauungsplan für die Kliniken und naturwissenschaftlichen Institute der Universität Heidelberg. In: PREUßISCHES FINANZMINISTERIUM (Hrsg.): Zentralblatt der Bauverwaltung. Jg.58. H.10. Preußisches Finanzministerium. Berlin. 247-256.

SCHMITT, A. (1986): Das Neuenheimer Feld nach 1945. In: RIEDL, P. A. (Hrsg.): Semper Apertus – Sechshundert Jahre Ruprecht-Karls-Universität Heidelberg 1386-1986. Bd.5. Springer. Heidelberg. 514-558.

SCHÜTZ, M. (2001): Geodaten gehen ins Netz – das multifunktionale Informationssystem der Hamburger Baubehörde in Intranet und Internet. In: Geo-Informations-Systeme / Geobit 5 (2001). 19-21.

SCHWARZ, M. (1996): Tradition verpflichtet – Die Universität als Bauherr. In: Stadt Heidelberg (Hrsg.): Architektur Forum Heidelberg. Architektur- und Wirtschaftsförderungsverlag. Heidelberg. 20-24.

SCHWICHTENBERG, H.; CONRAD, S.; GARTNER, T.; SCHEER, O. (2003): Windows Scripting lernen. Addison-Wesley. München.

SOMMER, H. (1998): Projektmanagement im Hochbau – Eine praxisnahe Einführung in die Grundlagen. Springer. Heidelberg.

SELLE, K. (2000): Was? Wer? Wie? Warum? – Voraussetzungen und Möglichkeiten einer nachhaltigen Kommunikation. Dortmunder Vertrieb für Bau- und Planungsliteratur. Dortmund.

SENATSVERWALTUNG FÜR STADTENTWICKLUNG BERLIN (2006): Pressemeldung – 3D-Stadtmodell Berlin vorgestellt. http://www.stadtentwicklung.berlin.de/aktuell/pressebox/archiv_volltext.shtml?arch_0605/nachricht2278.html. Abruf am 07.06.06.

STAATLICHE HOCHBAUVERWALTUNG BADEN-WÜRTTEMBERG (Hrsg.) (1994): Universität Heidelberg: Im Neuenheimer Feld – Dokumentation der Gesamtplanung. Universitätsbauamt Heidelberg. Neckarsulm.

STADT HEIDELBERG (Hrsg.) (2002): Stadtteilrahmenplan Neuenheim – Teil 2: Entwicklungskonzept und Maßnahmenvorschläge. Heidelberg.

STADT HEIDELBERG (2004): Beschlussvorlage des Gemeinderats – Straßenbahn im Neuenheimer Feld. Heidelberg. http://ww1.heidelberg.de/buergerinfo/vo0050.asp?__kvonr=13247&voselect=2864. Abruf am 02.03.07.

STADT HEIDELBERG (2005a): Informationsvorlage des Gemeinderats – Umbau der Kreuzungen Berliner Straße / Jahnstraße und Berliner Straße / Möchhofstraße. Heidelberg. http://ww1.heidelberg.de/buergerinfo/vo0050.asp?__kvonr=13783&voselect=2967. Abruf am 02.03.07.

STADT HEIDELBERG (2005b): Informationsvorlage des Gemeinderats - Umweltverträglichkeitsuntersuchung (UVU) 5. Neckarquerung mit Alternativen. Heidelberg. http://ww1.heidelberg.de/buergerinfo/vo0050.asp?__kvonr=13997&voselect=3317. Abruf am 02.03.07.

STEIDLER, F.; BECK, M. (2004): Cybercity Modeler, Generation, Updating and Continuation of 3D-Citiy Models with Online-Editing – Visualization with TerrainView 2.0. In: SCHRENK, M. (Hrsg.): CORP 2004 – Beiträge zum 9. Symposion zur Rolle der Informationstechnologie in der Stadt- und Raumplanung sowie zu den Wechselwirkungen zwischen realem und virtuellem Raum. Selbstverlag des Vereines CORP – Competence Center for Urban and Regional Development. Wien. 359-366.

STEIDLER, F.; BECK, M. (2005): CyberCity Modeler: Automatic Texturing of 3D City Models - TerrainView-Web: 3D Web-VRGIS. In: SCHRENK, M. (Hrsg.). CORP 2005 – Geo Multimedia 2005 - Beiträge zum 10. Symposion zur Rolle der Informations- und Kommunikationstechnologie in der Stadtplanung und Regionalentwicklung sowie zu den Wechselwirkungen zwischen realem und virtuellem Raum. Selbstverlag des Vereines CORP – Competence Center for Urban and Regional Development. Wien. 219-223.

STEPPAN, B. (2003): Einstieg in Java. Galileo Press. Bonn.

STRASSENBURG-KLECIAK, M. (2006): 3D-Navigation vor dem Durchbruch. In: GIS-Business. 12/2006. 10-11.

SUN (2007): Switching between the Microsoft VM and the Sun Java Runtime Environment (JRE). http://www.java.com/en/download/help/switchvm.xml. Abruf am 02.03.07.

THEIS, T. (2006): Einstieg in PHP 5 und MySQL. Galileo Press. Bonn.

TONNIER, A. (2002): Ergonomische Benutzerschnittstellen für GIS-basierte Informationssysteme für das WWW. Diplomarbeit. Geographisches Institut der Universität Heidelberg.

UNIVERSITÄT HEIDELBERG (Hrsg.) (1968): Verkehrsuntersuchung für das neue Universitätsgebiet. Heidelberg.

UNIVERSITÄTSBAUAMT HEIDELBERG (Hrsg.) (2002): Universität Heidelberg – Konzept für die Verkehrserschließung des Universitätsgebiets ´Im Neuenheimer Feld´. Heidelberg.

UNIVERSITÄTSBAUAMT HEIDELBERG (2003-2005): Pläne diverser Architekturbüros. DWG-Dateien.

UNIVERSITÄTSBAUAMT HEIDELBERG (2004a): Orientierungsplan. PDF-Datei.

UNIVERSITÄTSBAUAMT HEIDELBERG (2004b): Übersichtsplan Gesamtplanung 06/2004. DWG-Datei.

UNIVERSITÄTSBAUAMT HEIDELBERG (2005): Lageplan Bestand. DWG-Datei.

UNIVERSITÄTSKLINIKUM HEIDELBERG PLANUNGSGRUPPE MEDIZIN (2005): Ausbauplanung Heidelberger Klinikring. http://www.klinikum.uni-heidelberg.de/uploads/media/ Ausbau_Heidelberger_Ring.pdf. Abruf am 02.03.07.

VEPS (2005): Virtual Environmental Planning System. http://www.veps3d.org/site/files/28-jul-2005/16-57-52/VEPS_summary_05_dt_new.pdf. Abruf am 02.03.07.

VERMESSUNGSAMT HEIDELBERG (2002): Amtlicher Stadtplan Heidelberg auf CD-ROM. Heidelberg.

VERMESSUNGSAMT HEIDELBERG (2003): Digitaler Luftbildatlas Heidelberg auf CD-ROM. Heidelberg.

W3SCHOOLS (2007a): Browser and OS Platform Statistics. http://www.w3schools.com/browsers/browsers_stats.asp. Abruf am 02.03.07.

W3SCHOOLS (2007b): JavaScript Variables. http://www.w3schools.com/js/js_variables.asp. Abruf am 02.03.07.

WACKER, O. (1936): Der Baugedanke. In: SCHMIEDER, L.: Die neuen Kliniken und Naturwissenschaftlichen Institute der Universität Heidelberg. Brausdruck. Heidelberg. 2-6.

WATT, A. (2000): 3D Computer Graphics. Addison-Wesley. London.

WALSH, A. E.; BOURGES-SEVENIER, M. (2001): Web 3D. Prentice Hall. Upper Saddle River.

WELZEL, R.-W.; KLARNER, R. (2002): Realistische Modelle – Hamburger GIS-Lösung eröffnet neue Dimensionen. In: Geo-Informations-Systeme/ Geobit. 2/2002. 17-19.

WENZ, C. (2007): JavaScript und AJAX. Galileo Press. Bonn.

WERKLE, U. (1959): Die Universität Heidelberg: Der Bebauungsplan. In: STAATLICHE HOCHBAUVERWALTUNG NORDBADEN (Hrsg.): Staatliche Hochbauten Baden-Württemberg – Bauabteilung Nordbaden – Ein Jahrzehnt Wiederaufbau in Nordbaden 1948-1958. Kurz. Stuttgart. S.56-71.

WILK, C. (2007): Welt in Händen – Arbeiten mit Google Earth und World Wind. In: IX Special – Magazin. 01/2007. 69-76.

WOLGAST, E. (1986): Die Universität Heidelberg – 1386-1986. Springer. Heidelberg.

ZACH, C.; GRABNER, M.; SORMANN, M.; KARNER, K. (2005): Internet-basierte Echtzeitvisualisierung von 3D-Geodaten. In: COORS, V.; ZIPF, A. (Hrsg.): 3D-Geoinformationssysteme – Grundlagen und Anwendungen. Wichmann. Heidelberg. 202-213.

ZIPF, A. (2000): Deep Map, ein verteiltes historisches Touristeninformationssystem. Dissertation. Geographisches Institut der Universität Heidelberg.

ZIPF, A. (2007): GDI3D - Geodateninfrastruktur für 3D – Geodaten. http://webmap.geoinform.fh-mainz.de/gdi3d/index.htm. Abruf am 02.04.07.

Anhang

Die beiliegende CD enthält zwei Filmdateien, die die wesentlichen Ergebnisse dieser Arbeit präsentieren. Die Filme sind so ausgelegt, dass sie auf einem Standard-PC beispielsweise mit dem Windows Media Player abgespielt werden können. Zum Abspielen wird ein spezieller Codec benötigt. Sollte dieser nicht installiert sein, wird er bei bestehender Internetverbindung automatisch vom Windows Media Player heruntergeladen und installiert.

- **3D-Modell.avi**: Dieser Film zeigt das dreidimensionale Computermodell des Heidelberger Universitätscampus. Ausgewählte Areale des Modells werden in unterschiedlichen thematischen Zusammenhängen visualisiert.

- **3D-Informationssystem.avi**: Dieser Film zeigt ausgewählte Funktionalitäten und Inhalte des interaktiven, webbasierten 3D-Informationssystems. Diese Demonstration wurde online und in Echtzeit aufgezeichnet. Sie wurde nicht nachträglich bearbeitet. Die Methode, die für diese Aufzeichnung verwendet wurde, bringt systembedingt Mängel bezüglich der flüssigen Darstellung mit sich. Kleinere Bildaussetzer und stellenweise leicht stockende Darstellung sind auf diese Aufzeichnungsmethode und nicht auf Schwächen des in dieser Arbeit entwickelten 3D-Informationssystems zurückzuführen.

Erklärung

Ich erkläre hiermit, dass ich die vorgelegte Dissertation selbst verfasst und mich keiner anderen als der von mir ausdrücklich bezeichneten Quellen und Hilfen bedient habe.

Heidelberg, den 24.05.2007

Nicolai Freiwald